AMSTRAD

CPC 464

MANUALE D'USO



PERSONAL COMPUTER A COLORI CPC464 64K

Uso di questo manuale

L'uso dei calcolatori è progredito in un breve spazio di tempo. Tra tutti i progressi tecnologici del ventesimo secolo, l'uso dei calcolatori è senza dubbio il più stupefacente.

Le caratteristiche dell'hardware e del software sono migliorate così rapidamente che anche gli utenti stessi non sono stati in grado di seguirle. Per cercare di mostrare ai possessori del CPC464 tutte le potenzialità e le sottigliezze del suo BASIC, del suo sistema operativo e del suo hardware occorrerebbero migliaia di pagine.

Questa guida intende dunque essere una breve introduzione al CPC464 ed al suo software. Essa potrà essere affiancata da corsi di istruzione e pubblicazioni più specifici e dettagliati.

Gli utenti che già conoscono altri dialetti di BASIC apprenderanno rapidamente le caratteristiche più salienti del BASIC AMSTRAD. I principianti apprezzeranno invece la natura diretta e non ambigua della terminologia usata. Il BASIC AMSTRAD è stato infatti progettato appositamente per evitare le idiosincrasie che è possibile trovare in altre interpretazioni non standard del BASIC e per introdurre le fondamentali caratteristiche "real time" che non erano in precedenza disponibili su calcolatori di basso costo.

Il manuale è diviso in tre parti

Il primo è il Corso Fondamentale per Principianti, utile per introdurre i concetti del calcolo e la terminologia . Se non si è mai posseduto o usato un personal computer, si consiglia di partire proprio dal Corso Fondamentale.

Coloro che già possiedono una certa esperienza potranno cominciare dal Capitolo 1. Qui ripresentiamo un certo numero di argomenti riguardanti la predisposizione all'uso ed un pò di familiarizzazione. Ci siamo tuttavia concentrati nell'introduzione delle caratteristiche specifiche del CPC464 ed abbiamo fatto alcune assunzioni riguardo la familiarità dell'utente con la terminologia usata.

Ogni capitolo della prima parte fornisce una guida generale alle molte eccitanti caratteristiche del CPC464. Alcuni punti fondamentali vengono ripetuti per far meglio comprendere l'argomento - e poichè molti intenderanno passare direttamente alla grafica ed al suono dopo una brevissima introduzione all'uso della tastiera e degli aspetti più metodici del BASIC.

La "Guida al BASIC" Amstrad intende fornire una guida completa e ampiamente illustrata alla comprensione delle molteplici capacità del CPC464 e delle sue sconfinate possibilità nel campo dell'insegnamento, come console per videogiochi e, semplicemente, come computer; raccomandiamo così, per una conoscenza completa dell'argomento, di acquistarne una copia (sempre se già non è stato fatto).

Infine una estesa Appendice fornisce un'ampia panoramica dei concetti riguardanti il calcolo e riferimenti più specifici alla macchina.

Vi auguriamo ogni successo - non potreste aver scelto un computer con un miglior rapporto qualità prezzo, nè un computer con maggiori potenzialità nel campo dello sviluppo delle vostre conoscenze. Non vi è modo migliore per imparare ad usare un computer che usandone uno ed il CPC464 è particolarmente disponibile.



Il prodotto descritto in questo manuale ed i prodotti da utilizzare con esso sono soggetti ad un continuo sviluppo e miglioramento. Tutte le informazioni di natura tecnica ed i particolari riguardanti il prodotto ed il suo uso vengono forniti da AMSTRAD in buona fede. Tuttavia si riconosce che in questo manuale potrebbero esservi errori od omissioni. Raccomandiamo la compilazione e l'invio della scheda di registrazione e della garanzia.

AMSTRAD è lieta di ricevere commenti e suggerimenti relativi al prodotto o al manuale.

Ogni operazione di manutenzione e servizio deve essere effettuata a cura dei rivenditori autorizzati AMSTRAD. AMSTRAD non può accettare responsabilità per qualunque perdita o danno causato da operazioni di manutenzione e servizio effettuate da personale non autorizzato. Questo manuale intende semplicemente assistere il lettore all'uso del prodotto e quindi AMSTRAD non sarà responsabile per una qualunque perdita o danno derivanti dall'uso delle informazioni o a causa di un errore o omissione contenuti in questo manuale o ad un uso non corretto del prodotto.

All'interno di questo manuale il riferimento allo Z80 è usato grazie al riconoscimento della Zilog Inc.

Scritto da W. Poel, R. Perry, I. Spital, R.J. Watkins

Traduzione italiana di Giovanna Marsilio e Paolo Poli

Realizzato da Gruppo editoriale JCE - Via Ferri, 6 - 20092 Cinisello B. (MI)

Fotocomposizione: Lineacomp S.r.l. - Via Ferri, 6 - 20092 Cinisello B. (MI)

Pubblicato da AMSTRAD SpA

Prima edizione 1988

IMPORTANTE

Durante la lettura di questo manuale, stili di stampa differenti indicano i vari modi in cui vengono fatti riferimenti ai programmi, **[TASTI]** che non risultano nei caratteri stampati sul video e <descrizioni generali> associate al programma ma che non vanno inserite come parte di una istruzione.

1. Si colleghi il cavo di alimentazione sempre alla presa a 3 pin seguendo le istruzioni contenute nella prima parte intitolata 'Predisposizione'.

2. Non si colleghi mai la tastiera del computer, il monitor o l'alimentatore/modulatore a qualunque apparecchio o alimentatore che non sia descritto in questo manuale.
Non rispettando questa raccomandazione si potrebbero causare gravi danni e rendere nulla la garanzia.

3. Si tengano lontani dalla tastiera del computer, dal monitor e dall'alimentatore/modulatore i vasi di fiori, bicchieri di bibite, ecc. La presenza di liquidi all'interno di questi apparecchi potrebbe causare gravi danni. In questo caso si consulti del personale qualificato.

4. Non si chiudano o ricoprano le aperture di raffreddamento nella parte superiore e posteriore della tastiera del computer, del monitor e dell'alimentatore/modulatore.

5. Spegnendo il computer verrà perso tutto quanto si trovava nella memoria. Se si vuole salvare un programma, si veda il capitolo 2 dopo aver terminato il corso introduttivo.

6. Si raccomanda l'uso di cassette progettate appositamente per il computer. Tuttavia è possibile utilizzare cassette audio di buona qualità, non al Cromo o Metal e non più lunghe di 90 minuti (C90). Per permettere una localizzazione più rapida dei programmi memorizzati, si consiglia l'uso di cassette C12 (6 minuti per lato).

7. Cassette contenenti programmi per altri tipi di computer non potranno essere utilizzate con il CPC464.

8. Nel caso dalla cassetta che si sta usando siano state rimosse le linguette di sicurezza che evitano la cancellazione accidentale dei programmi, il tasto di registrazione non potrà essere premuto. Non si cerchi di forzare tale tasto; in questo caso si potrebbe danneggiare il meccanismo. Se si vuole ri-registrare una cassetta con le linguette di protezione rimosse, è sufficiente ricoprire le aperture sul retro della cassetta con nastro adesivo.

9. Ci si assicuri che il nastro abbia sorpassato la prima parte non magnetizzabile prima di cominciare a salvare un programma.

10. Non si deve utilizzare o lasciare nessuno degli apparecchi alla luce solare diretta, ad una temperatura eccessivamente calda o fredda, in aree soggette a polvere o a vibrazioni. Non si lascino le cassette vicino a campi magnetici, ad esempio vicino ad altoparlanti o grossi motori elettrici.

11. Una cura generale delle cassette ed una regolare pulizia dei meccanismi del registratore eviteranno errori nel salvataggio e nel ritrovamento dei programmi.

12. All'interno degli apparecchi non vi sono parti che possano essere manipolate da un utente. Non si cerchi perciò di aprire questi apparecchi. Ogni operazione di servizio dovrà essere effettuata da personale qualificato.

13. Non è possibile adattare o riprodurre in alcun modo in qualunque forma nè questo manuale nè parte di esso nè i programmi, nè i prodotti qui descritti.

Indice

Fondamenti 1: PREDISPOSIZIONE

Istruzioni iniziali per la predisposizione, la connessione e l'accensione del proprio sistema CPC464.

Il personal computer a colori CPC464 AMSTRAD puo' essere collegato ad uno dei seguenti dispositivi:

1.1 Un monitor a fosfori verdi GT65 AMSTRAD
1.2 Un monitor a colori CTM644 AMSTRAD
1.3 Un Modulatore/Alimentatore MP2 AMSTRAD e un ricevitore TV domestico.

Si prega di far riferimento alla sessione appropriata per la corretta installazione del proprio sistema e di seguire attentamente le istruzioni fornite.

1.1 Monitor a fosfori verdi GT65 AMSTRAD

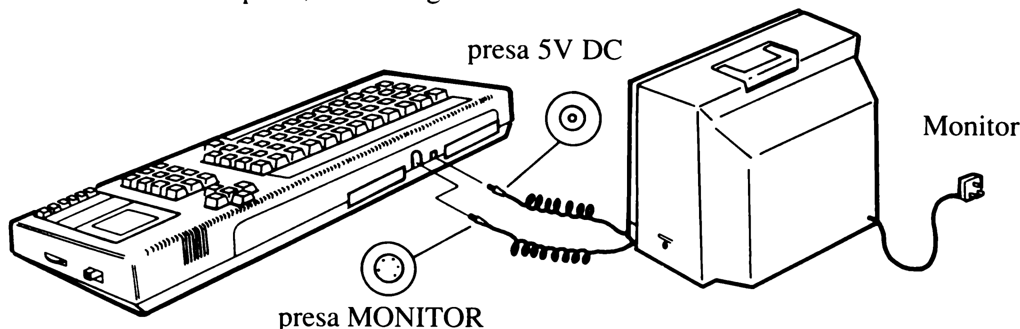
Disimballare il monitor e connettere la spina nella presa come segue:

ATTENZIONE

Quando il sistema non è in uso si estragga la spina dalla presa.

Non occorre fare connessioni interne quindi, non si cerchi di accedere all'apparecchiatura

Il computer deve essere posizionato di fronte al monitor su un tavolo vicino ad una presa di corrente. Si connetta il cavo con la presa più larga del monitor con la presa **MONITOR** situata sul retro del computer. Si connetta il capo della presa più piccola nella presa **5V DC** sul retro del computer; si veda figura 1.



Ci si assicuri che il pulsante di alimentazione **(POWER)** del monitor sia in posizione **OFF** (ovvero spento). Si connetta la spina del monitor alla presa di corrente.

Si accenda ora il monitor e successivamente il computer usando il commutatore **POWER** situato sulla destra di quest'ultimo.

L'indicatore rosso in alto al centro della tastiera deve essere acceso e il monitor visualizzerà:

Amstrad 64K Microcomputer (v1)
@1984 Amstrad Consumer Electronics plc
and Locomotive Software Ltd.
BASIC 1.0
Ready


Cursore

Per evitare di sottoporre gli occhi ad inutile tensione si regoli il controllo **BRIGHTNESS** (luminosità) affinchè la luminosità sia accettabile e non sia troppo abbagliante o troppo scura.

Occorre inoltre regolare il controllo **CONTRAST** (contrasto) al minimo possibile, secondo le proprie esigenze.

Il controllo verticale del GT65 contrassegnato **V-HOLD** deve essere regolato affinchè la visualizzazione sia correttamente posizionata in centro allo schermo, senza sfarfallii o rotazioni.

1.2 Un monitor a colori CTM644 AMSTRAD

Disimballare il monitor e connettere la spina nella presa come segue:

ATTENZIONE

Quando il sistema non è in uso si estragga la spina dalla presa.

Non occorre fare connessioni interne quindi, non si cerchi di accedere all'apparecchiatura interna.

Il computer deve essere posizionato di fronte al monitor su un tavolo vicino ad una presa di corrente. Si connetta il cavo con la presa più larga del monitor con la presa **MONITOR** situata sul retro del computer. Si connetta il cavo con la presa più piccola nella presa **5V DC** sul retro del computer; si veda la figura 1 nella precedente pagina.

Ci si assicuri che il pulsante di alimentazione **(POWER)** del monitor sia in posizione **OFF** (ovvero spento). Si connetta la spina del monitor alla presa di corrente.

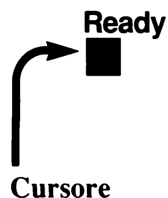
Si accenda ora il monitor e successivamente il computer usando il commutatore **POWER** situato sulla destra di quest'ultimo.

L'indicatore rosso in alto al centro della tastiera deve essere acceso e il monitor visualizzerà:

Amstrad 64K Microcomputer (v1)

@1984 Amstrad Consumer Electronics plc
and Locomotive Software Ltd.
BASIC 1.0

Ready



Cursore

Per evitare di sottoporre gli occhi ad inutile tensione si regoli il controllo **BRIGHTNESS** (luminosità) affinchè la luminosità sia accettabile e non sia troppo abbagliante o troppo scura.

1.3 Modulatore/Alimentatore MP2 AMSTRAD e ricevitore TV domestico.

MP2 è un oggetto addizionale che si potrebbe voler acquistare nel caso si stia usando il computer CPC464 con uno schermo a monitor verdi. L'MP2 permette di usare il computer con un normale televisore a colori e di visualizzare tutte le sfumature di colore del proprio CPC464.

Disimballare il monitor e connettere la spina nella presa come segue:

ATTENZIONE

Quando il sistema non è in uso si estragga la spina dalla presa.

Non occorre fare connessioni interne quindi, non si cerchi di accedere all'apparecchiatura interna.

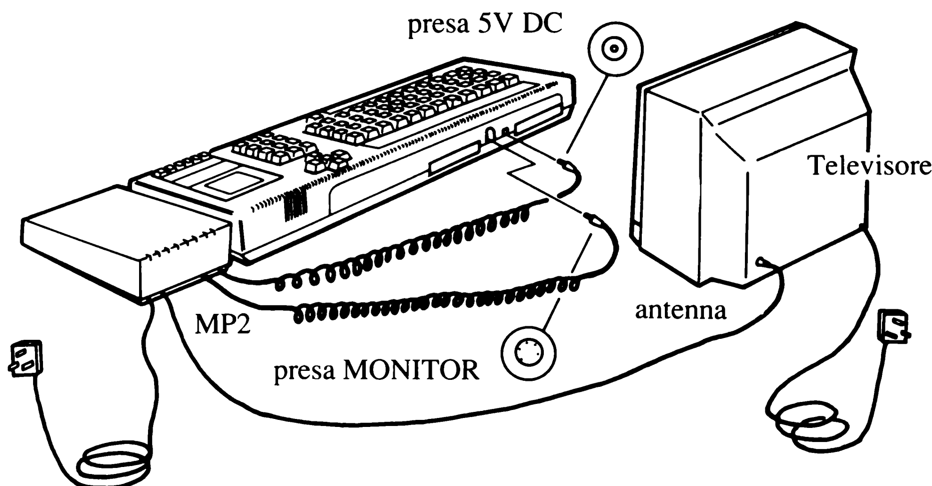


Figura 2: Modulatore MP2 e alimentatore.

Il modulatore (MP2) deve essere posizionato alla destra del computer su un tavolo vicino al televisore e ad una presa di corrente. Si connetta il cavo con la presa piu' larga di MP2 alla presa **MONITOR** situata sul retro del computer. Si connetta il cavo con la presa piu' piccola (DC) dell'MP2 alla presa **5V DC** sul retro del computer; si veda figura 2.

Si connetta la presa d'antenna dell'MP2 a quella del televisore.

Ci si assicuri che il pulsante di alimentazione **(POWER)** situato sulla destra del computer sia in posizione **OFF** (ovvero spento) si connetta la spina dell' MP2 alla presa di corrente.

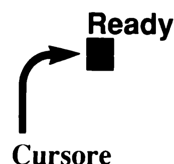
Si riduca il controllo di volume del televisore portandolo al minimo, si accenda il televisore e successivamente il computer usando il commutatore **POWER** situato sulla destra di questo.

L'indicatore rosso in alto al centro della tastiera deve essere acceso ed occorre sintonizzare il televisore al fine di ricevere il segnale dal computer.
Se si possiede un televisore con selezione di canali a pulsanti si selezioni un canale disponibile. Si predisponga il canale in modo che riceva il segnale dal computer seguendo le istruzioni del manuale d'uso del televisiore (il segnale sarà situato vicino al canale 36) fino a quando non si veda visualizzato il seguente messaggio:

Amstrad 64K Microcomputer (v1)

@1984 Amstrad Consumer Electronics plc
and Locomotive Software Ltd.
BASIC 1.0

Ready



Cursore

Si metta a punto la visualizzazione dello schermo in modo sia ben chiaro ciò che è scritto. Si dovrà ottenere una scritta gialla su sfondo blu.

Se la televisione che si possiede ha un leva rotante per la selezione dei programmi la si ruoti fino ad ottenere uno schermo chiaro e perfettamente fermo (verso il canale 36).

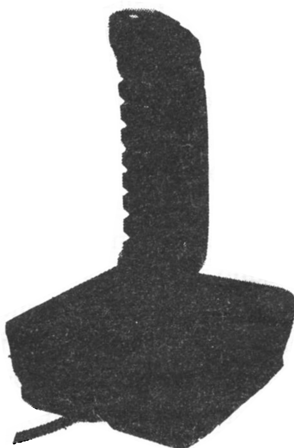
1.4 JOYSTICK

Il joystick AMSOFT modello JY1 è un oggetto addizionale che si può acquistare se si vuole utilizzare il CPC464 con giochi elettronici. Il joystick JY1 può essere collegato alla presa **USER PORTS (I/O)** situata sul retro del proprio computer. E' possibile, inoltre, collegare al CPC464 Amstrad due joystick. Il secondo joystick JY1 deve essere collegato nella presa del primo joystick.

SPINA PER IL JOYSTICK



JY1 JOYSTICK



PRESA PER IL SECONDO
JOYSTICK JY1



1.5 Cassetta Welcome

Insieme al computer, in uno dei sacchetti di politene, si sarà trovato la cassetta 'Welcome'. Si apra lo sportellino del Registratore premendo il pulsante **[STOP/EJECT]** e si inserisca la cassetta in esso come mostrato in figura 3; ci si assicuri che il Lato 1 **(SIDE 1)** sia rivolto verso l'alto:

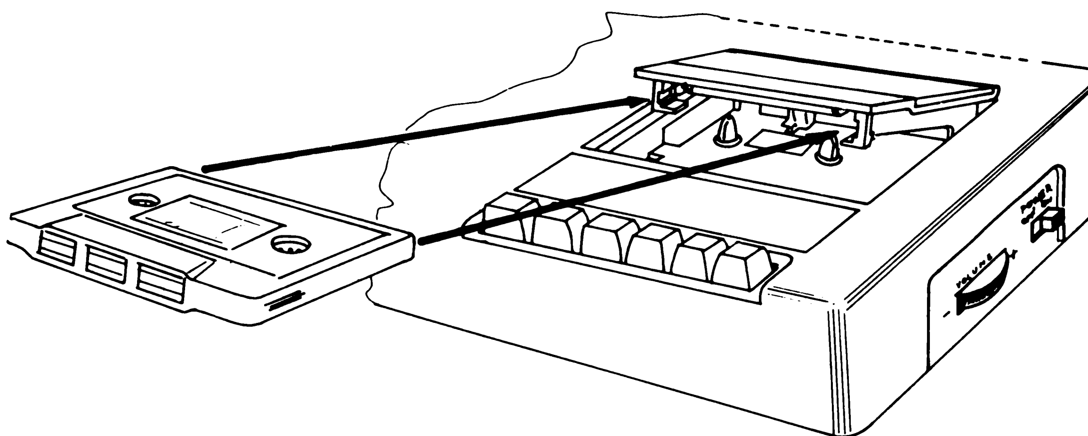


Figura 3: Inserimento corretto della cassetta nel registratore.

Si chiuda il coperchio, si prema il tasto **[REW]** del registratore per assicurarsi che il nastro sia completamente avvolto. Quando il nastro si ferma, si prema il tasto **[STOP/EJECT]**. Si azzeri il contanastro (000) premendo il pulsante **COUNTER RESET**.

Si mantenga premuto il tasto **[CTRL]** *E ALLO STESSO TEMPO* si prema il tasto **[ENTER]** situato di fianco al registratore in basso a destra del tastierino numerico. Sullo schermo verranno visualizzate le seguenti istruzioni:

RUN"
Press PLAY then any key:

Si prema ora il tasto **[PLAY]** situato davanti al registratore e successivamente una qualsiasi lettera, un qualsiasi numero oppure il tasto **[ENTER]** o la barra spazi.

Il nastro inizia a girare e poco dopo sullo schermo apparirà un messaggio:

Loading WELCOME 1 block 1

Occorrerà aspettare circa cinque minuti affinchè venga caricato il programma; si potrà vedere 'cosa accade' sullo schermo poichè il block cambierà in 2,3 ecc. fino a quando il nastro non si ferma: a questo punto il programma 'WELCOME' verrà eseguito. Ci si sieda e si guardi: il programma viene continuamente eseguito quindi, quando si è terminata la visione si prema due volte il tasto **[ESC].** Quest'ultimo serve a fermare l'esecuzione del programma; si può a questo punto premere il tasto **[STOP]** per estrarre la cassetta e girarla dall'altro lato.

Dopo aver girato la cassetta sul lato 2 ci si ricordi ancora di premere il tasto **[REW]** per assicurarsi che il nastro sia completamente avvolto.

Si mantenga premuto il tasto **[CTRL]** *E ALLO STESSO TEMPO* si prema il tasto **[ENTER]** situato di fianco al registratore in basso a destra del tastierino numerico. Sullo schermo verranno visualizzate le seguenti istruzioni:

RUN"
Press PLAY then any key:

Si prema ora il tasto **[PLAY]** situato davanti al registratore e successivamente una qualsiasi lettera, un qualsiasi numero oppure il tasto **[ENTER]** o la barra spazi.

Il nastro inizia a girare e poco dopo sullo schermo apparirà un messaggio:

Loading WELCOME 2 block 1

Si seguano a questo punto le istruzioni fornite sullo schermo: il programma inviterà l'utente a partecipare facendo scrivere alcune istruzioni.

1.6 COME CARICARE ALTRE CASSETTE

IL NASTRO WELCOME PUO' ESSERE SOLO CARICATO ED ESEGUITO come descritto nel precedente paragrafo (1.5). I programmi BASIC non protetti possono essere caricati seguendo due metodi alternativi. Si prema il tasto **[REW]** per far avvolgere completamente il nastro della cassetta: al termine premere immediatamente il tasto **[STOP/EJECT]**.

Si riavvi il computer, pulendo la sua memoria, premendo i tasti **[CTRL]**, **[SHIFT]** e **[ESC]** in ordine mantenendo premuti i primi due fino a quando il tasto **[ESC]** non è stato completamente premuto. Lo schermo verrà pulito e verrà visualizzato lo stesso messaggio che appare quando si accende il computer.

L'espressione **[ENTER]** delle istruzioni che seguono implica che occorre premere uno dei due tasti contrassegnati **[ENTER]** e non scrivere la parola ENTER! Le " si ottengono mantenendo premuto il tasto **[SHIFT]** e premendo il tasto **2**.

Si scriva:

Load "" **[ENTER]**

Il computer chiede:

Press PLAY then any key:

Si prema ora il tasto **[PLAY]** situato davanti al registratore e successivamente una qualsiasi lettera, un qualsiasi numero oppure il tasto **[ENTER]** o la barra spazi.

Il nastro inizia a girare e poco dopo sullo schermo apparirà un messaggio:

Loading *<nome programma>* block 1

Il numero del block verrà continuamente incrementato fino a quando non ha fine il caricamento del nastro e viene visualizzato sullo schermo il seguente messaggio:

Ready

In alternativa è possibile specificare il nome del programma che si desidera caricare. Per far ciò si scriva:

Load *"<titolo>"* **[ENTER]**

Il computer richiede di:

Press PLAY then any key:

Si prema ora il tasto **[PLAY]** situato davanti al registratore e successivamente una qualsiasi lettera, un qualsiasi numero oppure il tasto **[ENTER]** o la barra spazi.

Il nastro inizia a girare. Se il programma richiesto non si trova all'inizio del nastro il computer lo ricercherà. Si faccia attenzione a scrivere il titolo esatto poichè il computer cercherà fino a quando non trova quello equivalente.

Se, mentre il computer ricerca il titolo desiderato, trova un titolo differente da quello inserito visualizzerà un messaggio sullo schermo:

Found *<altro titolo>* block 1

Il computer non caricherà tale programma ma continuerà a cercare nel nastro il titolo esatto fino a quando non lo trova o l'utente non prema il tasto **[ESC]** per fermare la ricerca.

Quando il programma è stato trovato sullo schermo apparirà il seguente messaggio:

Loading *<titolo>* block 1

Il numero del block verrà continuamente incrementato fino a quando non ha fine il caricamento del nastro e viene visualizzato sullo schermo il seguente messaggio:

Ready

Successivamente si scriva:

run **[ENTER]**

Per eseguire direttamente il programma senza richiedere precedentemente il suo caricamento si scriva semplicemente:

RUN "" **[ENTER]**

il computer rispondera':

Press PLAY then any key:

Dopo aver premuto **[PLAY]**, seguito da una lettera, un numero, la barra spaziatrice o il tasto **[ENTER],** il computer cercherà, caricherà ed eseguirà il programma senza che ci sia bisogno di immettere altre istruzioni dalla tastiera. Si può interrompere questo procedimento premendo, come di consueto, il tasto **[ESC]**.

1.7 Come caricare cassette pre-registrate

Le istruzioni fornite fino a questo momento permettono di caricare molti programmi disponibili per il CPC464.

Per il caricamento si faccia comunque riferimento anche alle istruzioni stampate su ogni pacchetto software.

1.8 Come salvare un programma

Dopo aver usato un programma questo può essere salvato (registrato). Si inserisca correttamente la cassetta e si chiuda lo sportellino (non deve essere stata eliminata la protezione dalla registrazione della cassetta). Si prema il tasto **[REW]** per avvolgere il nastro completamente, ricordandosi successivamente di premere il tasto **[STOP/EJECT]** per fermarlo. Si scriva:

save *"<titolo programma>"* **[ENTER]**

Il computer risponde:

Press REC and PLAY then any key:

Si premano ora i tasti **[REC]** e **[PLAY]** del registratore e successivamente si prema un tasto qualsiasi (lettera, numero, spazio o il tasto **[ENTER]).**

Il computer risponde:

Saving *<titolo programma>* block 1

Quando il programma è stato salvato il nastro della cassetta si fermerà e verrà visualizzato sullo schermo il messaggio: Ready. Si prema ora il tasto **[STOP/EJECT]**: il programma e' stato salvato.

Si noti che non è possibile salvare programmi e giochi pre-registrati su proprie cassette vergini.

Tali programmi infatti sono protetti contro copie non autorizzate.

Fondamenti 2: CONOSCENZA DELLA TASTIERA

Spiegheremo ora la funzione di alcuni tasti del computer. Coloro che già conoscono l'utilizzo di un computer possono saltare questa parte.

[ENTER]

Vi sono due tasti **[ENTER]**. Entrambi questi tasti immettono informazioni battute al computer. Dopo aver premuto il tasto **[ENTER]** viene presentata sullo schermo una nuova linea. Ogni istruzione immessa nel computer dovrà essere seguita dalla pressione del tasto **[ENTER].**

Da ora in poi quando si troverà **[ENTER]** alla fine di una istruzione o di una linea di programma, occorrerà premere il tasto **[ENTER].**

[DEL]

Questo tasto cancella il carattere (lettera o numero) a sinistra del cursore sullo schermo. Si immetta abcd e si vedrà che la lettera d è posta alla sinistra del cursore. Se si decide di eliminare la lettera d, si prema il tasto **[DEL]** e la lettera d sparirà. Premendo e mantenendo premuto il tasto **[DEL]** verranno cancellate anche le lettere abc.

[SHIFT]

Vi sono due tasti **[SHIFT].** Premendo uno di questi tasti e mantenendolo premuto mentre si preme un altro tasto, apparirà sullo schermo una lettera maiuscola o il simbolo superiore del tasto.

Si batta la lettera **a** e poi si mantenga premuto il tasto **[SHIFT]** e si batta nuovamente la lettera **a**. Sullo schermo appariranno le lettere:

aA

Ora si inseriscano alcuni spazi, mantenendo premuta la barra spaziatrice. Si provi ora a fare la stessa operazione utilizzando i tasti numerici della prima linea della tastiera, sopra i tasti delle lettere. Si batta il numero 2 e poi, tenendo premuto il tasto **[SHIFT]** si prema nuovamente il tasto 2. Sullo schermo apparirà:

2"

Ora che si sa cosa avviene tenendo premuto il tasto **[SHIFT]** e premendo un altro tasto, si provi ad immettere altri caratteri, da soli o in congiunzione con il tasto **[SHIFT]**.

[CAPS LOCK]

Questo tasto ha un comportamento simile al tasto **[SHIFT]** tranne per il fatto che occorre premerlo una sola volta. Da quel momento in poi, ogni lettera immessa verrà visualizzata in maiuscolo ma il funzionamento della parte della tastiera dedicata ai numeri non verrà modificato. Si prema **[CAPS LOCK]** e poi:

abcdef123456

Sullo schermo si noterà che, sebbene le lettere appariranno in maiuscolo, i numeri non sono stati tramutati in simboli. Se occorresse immettere un simbolo mentre è in azione il tasto **[CAPS LOCK]** è sufficiente premere il tasto **[SHIFT]** prima di premere il tasto. Si premano i seguenti tasti mentre si tiene premuto il tasto **[SHIFT]**:

ABCDEF123456

Sullo schermo apparirà la scritta:

ABCDEF!"#$%&

Se si intende ritornare all'uso delle lettere minuscole, si prema nuovamente il tasto **[CAPS LOCK]**.

Se si vogliono inserire lettere maiuscole e i simboli della parte superiore del tasto senza dover continuamente premere il tasto **[SHIFT]**, è possibile, tenendo premuto il tasto **[CTRL]**, premere il tasto **[CAPS LOCK]**. Ora si inserisca:

abcdef123456

Sullo schermo verrà visualizzato:

ABCDEF!"#$%&

Per inserire numeri dopo la pressione dei tasti **[CTRL]** e **[CAPS LOCK]** si possono utilizzare i tasti numerici sulla destra della tastiera.

Tenendo premuto il tasto **[CTRL]** e premendo il tasto **[CAPS LOCK]** si ritornerà alla modalità in cui si era precedentemente (blocco maiuscole o minuscole). Se si fosse ritornati in stato di blocco maiuscole, premendo nuovamente il tasto **[CAPS LOCK]** si ritornerà in modalità minuscole.

[CLR]

Questo tasto è usato per cancellare il carattere su cui si trova il cursore.

Si immetta ABCDEFGH. Il cursore si troverà alla destra dell'ultimo carattere inserito (H). Si prema ora quattro volte il tasto di cursore a sinistra [←]. Il cursore si sposterà a sinistra di quattro posizioni e si troverà sulla lettera E.

Si noti che la lettera E è ancora visibile sotto il cursore. Si prema una volta il tasto **[CLR]** e la lettera E sparirà mentre le lettere FGH si sposteranno di una posizione a sinistra; la lettera F si troverà sotto al cursore. Premendo e tenendo premuto il tasto **[CLR]** verranno cancellate le lettere FGH.

[ESC]

Questo tasto serve per uscire da una funzione che il computer sta eseguendo. Premendo una volta il tasto **[ESC]** la funzione effettuata dal computer viene momentaneamente interrotta per riprendere alla pressione di un qualunque altro tasto.

Premendo due volte il tasto **[ESC]** la funzione in corso di esecuzione sul computer viene invece definitivamente interrotta. Il computer sarà quindi pronto ad eseguire qualunque istruzione verrà immessa.

Si prema il tasto **[ESC]** due volte.

IMPORTANTE

Quando si raggiunge il limite destro dello schermo perchè si sono inseriti più di 40 caratteri, il carattere successivo apparirà sul bordo sinistro della riga successiva. Ciò significa che **NON** occorre premere **[ENTER]** come con una macchina per scrivere. Il computer effettuerà questa operazione automaticamente e risponderà ad un eventuale **[ENTER]** con un messaggio di errore (in genere Syntax error).

Errori di sintassi

Se sullo schermo appare il messaggio Syntax error, significa che il computer non comprende una istruzione immessa. Ad esempio, si inserisca:

printt **[ENTER]**

Nello schermo apparirà il messaggio:

Syntax error
(ovvero: Errore di sintassi)

Il messaggio appare perchè il computer non ha compreso l'istruzione printt. Se si commette un errore in una linea di programma, come in:

10 printt "abc" **[ENTER]**

Il messaggio Syntax error non apparirà finchè l'istruzione non verrà esaminata dal computer durante l'esecuzione del programma.

Si immetta:

run **[ENTER]**

Sullo schermo apparirà:

```
Syntax error in 10
10 printt "abc"
```

Questo messaggio indica la linea in cui è capitato l'errore e visualizza la linea di programma con il cursore per permettere di correggerla.

Si prema il tasto "cursore a destra" [->] fino a portare il cursore sulla lettera t. Poi si prema il tasto **[CLR]** per cancellare la lettera ed il tasto **[ENTER]** per immettere la linea corretta.

Si immetta:

run **[ENTER]......** e si potrà notare che il computer ha accettato l'istruzione e ha stampato abc.

Introduzione alle parole chiave del BASIC AMSTRAD.

Nel capitolo 8 viene riportata una descrizione illustrata di tutte le parole chiave del BASIC utilizzate dal BASIC AMSTRAD. In questa parte introduciamo alcune delle parole chiave più usate.

CLS

Si immetta: cls (cancella schermo). E' possibile usare sia le lettere minuscole che le maiuscole. Poi si prema **[ENTER]**. Lo schermo verrà cancellato e nell'angolo in alto a sinistra dello schermo apparirà la parola Ready (Pronto) ed il cursore.

PRINT

Si usa questa parola chiave quando si vogliono stampare sullo schermo caratteri, parole o cifre. Si immetta la seguente linea:

print "ciao" **[ENTER]**

Sullo schermo apparirà:

ciao

Le virgolette indicano ciò che deve essere stampato. ciao apparirà sullo schermo non appena si premerà il tasto **[ENTER]**. Si immetta ora cls **[ENTER]** per cancellare lo schermo.

RUN

L'esempio precedente mostrava un programma di una sola linea. Ma i programmi hanno normalmente più di una linea. All'inizio di ogni riga vi è un numero. Questi numeri dicono al computer in quale ordine si deve eseguire il programma. Quando si preme il tasto **[ENTER]** la linea viene memorizzata dal computer fino all'esecuzione del programma. Si immetta:

10 print "ciao"**[ENTER]**

Si noterà che alla pressione del tasto **[ENTER]** *non* verrà stampato ciao sullo schermo. Occorre infatti immettere la parola run. Si immetta:

run **[ENTER]**

Ora apparirà la parola ciao sullo schermo. Al posto di print è possibile usare il simbolo ?, come in:

10 ? "ciao" **[ENTER]**

LIST

Dopo che un programma è stato memorizzato, è possibile verificare ciò che è stato inserito vedendo il listato del programma. Si immetta:

list **[ENTER]**

e sullo schermo apparirà:

10 PRINT "ciao"

che è il programma memorizzato.

Si noterà che la parola PRINT è in maiuscolo. Ciò significa che il computer ha riconosciuto PRINT come parola chiave del BASIC.

Si immetta ora cls **[ENTER]** per cancellare lo schermo. Sebbene lo schermo venga cancellato, il programma resta nella memoria del computer.

GOTO

La parola chiave GOTO dice al computer di "saltare" da una linea all'altra, per evitare l'esecuzione di un certo numero di linee o per formare un ciclo. Si immetta:

10 print "ciao" **[ENTER]**
20 goto 10 **[ENTER]**

Poi:

run **[ENTER]**

e la parola ciao verrà scritta ripetutamente sulla sinistra dello schermo. Per fermare l'esecuzione del programma, si prema il tasto **[ESC]**. Per riprenderla si prema un qualunque altro tasto. Per fermare definitivamente l'esecuzione, si prema due volte il tasto **[ESC]**.

Si immetta:

cls **[ENTER]**

per cancellare lo schermo.

Per vedere le parole ciao stampate una in fianco all'altra, linea per linea, fino al riempimento completo dello schermo, si utilizzi lo stesso programma ma con un punto e virgola (;) dopo le ultime virgolette ("). Si immetta:

```
10 print "ciao"; [ENTER]
20 goto 10 [ENTER]
run [ENTER]
```

Si sarà notato che il punto e virgola indica al computer di stampare il prossimo carattere immediatamente dopo il precedente. Si termini il programma premendo due volte il tasto **[ESC]**. Ora si reimmetta la linea 10 con una virgola (,) al posto del punto e virgola (;).

```
10 print "ciao", [ENTER]
run [ENTER]
```

Si noterà che la virgola indica al computer di stampare il prossimo carattere (o gruppo di caratteri) dopo 13 posizioni dall'inizio del precedente. Questa possibilità è utile per visualizzare informazioni su colonne. Si noti però che se un gruppo di caratteri è più lungo di 12 caratteri, il gruppo successivo verrà stampato dopo altri 13 caratteri.

Di nuovo, per uscire dall'esecuzione del programma si prema due volte il tasto **[ESC]**. Per cancellare completamente la memoria del computer, si tengano premuti i tasti **[SHIFT][CTRL]** ed **[ESC]** esattamente in quest'ordine, ed il computer si reinizializzerà.

INPUT

Questo comando serve per far sapere al computer che deve essere immesso qualcosa, ad esempio la risposta ad una domanda. Si immettano le linee:

```
10 input "quanti anni hai"; eta [ENTER]
20 print "sembri piu' giovane di ";eta;" anni." [ENTER]
run [ENTER]
```

Sullo schermo apparirà:

```
quanti anni hai?
```

Si immetta la propria età e si prema **[ENTER]**. Se la vostra età era di 18 anni, lo schermo mostrerà:

```
sembri più giovane di 18 anni.
```

Questo esempio mostra l'uso del comando input con una variabile numerica. La parola eta stata messa alla fine della linea 10 e quindi il computer ha associato la parola eta con il numero immesso ed ha poi stampato questo numero al posto della parola eta nella linea 20. Così come è stata stata usata l'espressione eta per una variabile numerica, avremmo potuto anche usare una sola lettera, ad esempio b.

Si reinizializzi il computer per cancellare la memoria. (Tasti **[CTRL][SHIFT]** e **[ESC]**). Se si fosse voluto effettuare un input di caratteri (lettere o lettere e numeri) si sarebbe dovuto usare un simbolo di dollaro al termine del nome della variabile. Si inserisca il seguente programma: (Nella linea 20 occorrerà inserire uno spazio dopo la o di ciao ed uno prima della i di il)

10 input "Qual è il tuo nome"; nome$ **[ENTER]**
20 print "Ciao";nome$ il mio nome è Arnold" **[ENTER]**
run **[ENTER]**

Sullo schermo apparirà:

Qual'è il tuo nome?

Si immetta il proprio nome seguito da **[ENTER]**

Se il nome immesso era Federico, sullo schermo apparirà:

Ciao Federico il mio nome è Arnold

Interesserà forse sapere che Arnold era il nome in codice del CPC464 durante le fasi di sviluppo. Così come è stata usata l'espressione nome$ per una variabile stringa, si sarebbe potuto anche usare una sola lettera, ad esempio a$. Uniamo ora i due esempi per formare un unico programma.

Si reinizializzi il computer premendo **[CTRL][SHIFT]** e **[ESC]** e poi si immettano le seguenti linee:

5 cls **[ENTER]**
10 input "Come ti chiami"; a$ **[ENTER]**
20 input "Quanti anni hai"; b **[ENTER]**
30 print "Devo dire ";a$ " che non dimostri"; b"anni" **[ENTER]**
run **[ENTER]**

In questo programma abbiamo usato 2 variabili: a$ per il nome e b per l'età. Sullo schermo apparirà:

Come ti chiami?

Ora si inserisca il proprio nome (ad es. Federico) e poi **[ENTER]**. Verrà chiesto:

Quanti anni hai?

Ora si inserisca la propria età (ad es. 18) e poi **[ENTER]**. Se il nome inserito era Federico e l'età 18, sullo schermo apparirà:

Devo dire Federico che non dimostri 18 anni

CORREZIONE DI UN PROGRAMMA

Se una qualunque delle linee del programma era stata immessa non correttamente ed ha prodotto un messaggio Syntax error o un altro messaggio di errore, è possibile correggere la linea invece di riscriverla. Per provare questa operazione, si inseriscano queste linee di programma non corrette:

5 clss **[ENTER]**
10 input "Come t chiami"; a$ **[ENTER]**
20 input "Quanti anni hai"; b **[ENTER]**
30 print "Devo dire";a$ " che non dimostri"; b"anni" **[ENTER]**

Nel programma vi sono 3 errori:
Nella linea 5 abbiamo inserito clss al posto di cls.
Nella linea 10 abbiamo inserito t invece di ti.
Nella linea 30 abbiamo dimenticato uno spazio tra dire e le virgolette (").

Vi sono 3 modi per correggere un programma. Il primo consiste nella ribattitura della linea. Quando una linea viene ribattuta e immessa, essa sostituisce la linea con lo stesso numero presente nella memoria del computer. Il secondo metodo usa la parola *edit* ed infine vi è il metodo copia-cursore.

METODO EDIT

Per correggere l'errore alla linea 5, si immetta:

edit 5 **[ENTER]**

La linea 5 verrà stampata con il cursore posizionato sul 5. Per cancellare la s in più di clss, si prema il tasto di cursore a destra [->] fino a portare il cursore sull'ultima lettera s e poi si prema il tasto **[CLR]**. La s sparirà. Ora si prema **[ENTER]** e la linea 5 sarà corretta anche in memoria. Si immetta:

list **[ENTER]**

e si vedrà che la linea 5 è stata corretta.

METODO CURSORE COPIA

Per correggere gli errori alle linee 10 e 30, si tenga premuto il tasto **[SHIFT]** e poi si prema il tasto cursore in alto [↑] finchè il cursore copia si trova all'inizio della linea 10. Si noterà che il cursore principale non si è mosso e che vi sono ora 2 cursori sullo schermo. Ora si prema il tasto **[COPY]** finchè il cursore copia si trova nello spazio tra t e chiami. Si noterà che la linea 10 verrà riscritta sull'ultima linea e che il cursore si ferma nella stessa posizione del cursore copia. Ora si batta la lettera i. Essa apparirà solo nell'ultima linea.

Il cursore principale si è spostato mentre il cursore copia è rimasto dove era. Ora si prema il tasto **[COPY]** finchè tutta la linea 10 non sarà stampata. Si prema **[ENTER]** e la linea 10 verrà memorizzata. Il cursore copia sparirà ed il cursore principale si porterà sulla nuova linea 10. Per correggere il secondo errore, si tenga premuto il tasto **[SHIFT]** e si prema il tasto cursore in alto [↑] per portare il cursore copia all'inizio della linea 30.

Si prema ora il tasto **[COPY]** finchè il cursore copia si trova sulle virgolette vicine alla parola dire. Ora si prema una volta la barra spaziatrice. Sulla linea inferiore apparirà uno spazio. Si tenga premuto il tasto **[COPY]** finchè il resto della linea 30 non verrà stampato. Poi si prema **[ENTER]**. Si può ora listare il programma per controllare che le correzioni siano state fatte immettendo: list **[ENTER]**

Ora si reinizializzi il computer premendo i tasti **[CTRL][SHIFT]** e **[ESC]**.

IF THEN

Estenderemo ora il precedente programma con l'uso dei comandi IF (SE) e THEN (ALLORA).
Si inserisca il seguente programma; si noti che vi sono due nuovi simboli. < significa "minore di" e si trova vicino alla lettera **M**, > significa "maggiore di" e si trova in fianco al tasto < (minore di).

```
5 cls [ENTER]
10 input "Come ti chiami"; a$ [ENTER]
20 input "Quanti anni hai"; anni [ENTER]
30 if anni < 13 then 60 [ENTER]
40 if anni < 20 then 70 [ENTER]
50 if anni > 19 then 80 [ENTER]
60 print "Dunque "a$" sei quasi un ragazzo a"anni"anni":end [ENTER]
70 print "Dunque "a$" sei un ragazzo a"anni"anni":end [ENTER]
80 print "Dunque "a$" non sei più un ragazzo a"anni"anni" [ENTER]
```

Per controllare che il programma sia corretto, si immetta:

list **[ENTER]**

Ora si immetta:

run **[ENTER]**

Ora si risponda alle domande presentate dal computer e si veda ciò che succede. Si capirà ora l'effetto dei comandi if e then all'interno di un programma. Abbiamo anche introdotto la parola end alla fine delle linee 60 e 70. La parola chiave end (fine) viene usata per fermare l'esecuzione del programma. Se alla linea 60 non vi fosse stato un end, il programma avrebbe continuato la propria esecuzione con le linee 70 e 80.

Nello stesso modo se non vi fosse stato un end alla linea 70 il programma avrebbe continuato la propria esecuzione con la linea 80. I due-punti (:) prima della parola end la separa dall'istruzione precedente. I due-punti servono per porre due o più istruzioni in una linea di comando. La linea 5 all'inizio del programma serve per cancellare lo schermo.

Altre parole chiave associate ad IF e THEN sono ELSE, OR e GOTO. L'uso di tali parole in un comando IF diventerà più chiaro mano a mano che si procederà nella lettura di questo manuale. Si reinizializzi il computer premendo i tasti **[CTRL][SHIFT]** e **[ESC]**.

FOR TO NEXT

Vedremo ora l'uso dei comandi FOR, TO e NEXT. In questo esempio faremo stampare al computer i multipli di 12 (1x12, 2x12, 3x12, ecc.). Si immetta il seguente programma; il simbolo * significa moltiplicazione.

```
5 cls [ENTER]
10 for a = 1 to 20 [ENTER]
20 print a" * 12 ="a *12 [ENTER]
30 next a [ENTER]
run
```

Si noterà che non tutti i numeri saranno incolonnati; si provi dunque ad inserire quest'altro programma:

```
5 cls [ENTER]
10 for a = 1 to 9 [ENTER]
20 print a" * 12 ="a *12 [ENTER]
30 next a [ENTER]
40 for a = 10 to 20 [ENTER]
50 print a" * 12 ="a *12 [ENTER]
60 next a [ENTER]
run
```

Si provi questo programma per il calcolo dei multipli anche con altri numeri, ad esempio per i multipli di 17, sostituendo al 12 delle linee 20 e 50 il numero 17. Infine si reinizializzi il computer premendo i tasti **[CTRL][SHIFT]** e **[ESC]**. Si noti che è possibile specificare un passo per il ciclo FOR TO NEXT usando il comando STEP. Per ulteriori informazioni, si studi la descrizione di FOR nel capitolo 8.

ARITMETICA SEMPLICE

Il CPC464 può anche essere facilmente usato come calcolatrice.

Per comprendere ciò si vedano gli esempi seguenti. In questo paragrafo useremo il simbolo ? al posto della parola chiave PRINT. Le risposte verranno stampate immediatamente dopo che sarà stato premuto il tasto **[ENTER]**.

ADDIZIONE

(per il più si usa **[SHIFT]** e ;)

Si immetta:

?3+3 **[ENTER]**
6
(non deve essere inserito il simbolo di uguale (=))

Si immetta:

?8+4 **[ENTER]**
12

SOTTRAZIONE

(per il meno si usa il tasto con il simbolo di =)

Si immetta:

?4-3 **[ENTER]**
1

Si immetta:

?8-4 **[ENTER]**
4

MOLTIPLICAZIONE

(per il per si usa **[SHIFT]** e : in quanto * sta per x)

Si immetta:

?3*3 **[ENTER]**
9

Si immetta:

?8*4 **[ENTER]**
32

DIVISIONE

(per il diviso si usa il tasto che riporta anche ? in quanto / sta per :)

Si immetta:

?3/3 **[ENTER]**
1

Si immetta:

?8/4
2

RADICE QUADRATA

Per trovare la radice quadrata di un numero si usa sqr( ). Il numero di cui si vuole la radice quadrata dovrà essere posto tra le due parentesi.

Si immetta:

?sqr(16) **[ENTER]** (significa $\sqrt{16}$)
4

Si immetta:

?sqr(100) **[ENTER]**
10

ESPONENZIALE

(si usi il tasto che riporta il simbolo di £) L'esponenziale è l'elevamento a potenza di un numero. Ad esempio 3 al quadrato (3^2) , 3 al cubo (3^3), ecc.

Si immetta:

?3↑3 **[ENTER]** (significa 3^3)
27

Si immetta:

?8↑4 **[ENTER]** (significa 8^4)
4096

RADICE CUBICA

Si può facilmente calcolare la radice cubica di un numero usando un metodo simile al precedente. Per trovare la radice cubica di 27, si immetta:

?27↑(1/3) **[ENTER]**
3

Per trovare la radice cubica di 125, si immetta:

?125↑(1/3) **[ENTER]**
5

CALCOLI MISTI (+,-,*,/,MOD,\)

I calcoli misti sono compresi dal computer ma calcolati secondo certe priorità. La massima priorità è data alla moltiplicazione ed alla divisione, poi all'addizione ed alla sottrazione. Queste priorità si applicano a calcoli contenenti solo queste quattro operazioni. Le priorità saranno ulteriormente spiegate più avanti e comprenderanno l'esponenziale, ecc.

Se il calcolo era:

```
3+7-2*7/4
```

Si potrebbe pensare venga calcolato in questo modo:

```
3+7-2*7/4
=  8*7/4
=   56/4
=     14
```

Mentre verrà calcolato così:

```
3+7-2*7/4
=3+7-14/4
=3+7-3.5
=10-3.5
=6.5
```

Infatti immettendo la seguente istruzione:

?3+7-2*7/4 **[ENTER]**

la risposta sarà: 6.5

Si può cambiare il modo in cui il computer effettua questo calcolo utilizzando delle parentesi. Il computer effettuerà i calcoli contenuti nelle parentesi prima di passare a quelli fuori dalle parentesi. Si dimostri ciò immettendo il calcolo sopra presentato ma utilizzando le parentesi.

Si immetta:

?(3+7-2)*7/4 **[ENTER]**
14

ALTRI ESPONENTI

Se nei propri calcoli si devono usare numeri molto grandi o molto piccoli, talvolta è utile usare la notazione scientifica. La lettera E indica l'elevamento a potenza della base 10 ad un numero. Si può usare la lettera e in maiuscolo o in minuscolo.

Ad esempio 300 è la stessa cosa di $3x10^2$. In notazione scientifica perciò si scriverà 3E2. In modo simile 0.03 è la stessa cosa di $3x10^2$. Si provino i seguenti esempi:

1. 30x10

Si può immettere:

?30*10 **[ENTER]**
300

oppure:

?3e1*1e1 **[ENTER]**
300

2. 3000x1000

Si immetta:

?3e3*1e3 **[ENTER]**
3000000

3. 3000x0.001

Si immetta:

?3e3*1e-3 **[ENTER]**
3

Fondamenti 3: Grafica, modalità e suono

Il CPC464 Colour Personal Computer Amstrad possiede tre modalità di visualizzazione sullo schermo: Mode 0, Mode 1 e Mode 2.

Quando il computer viene acceso esso si trova automaticamente in Mode 1.

Per capire la differenza tra queste modalità si accenda il computer e si prema il tasto 1. Lo si mantenga premuto fino a quando non si sono riempite due linee di 1. Se si conta ora il numero di 1 presenti su una linea si vedrà che questo è pari a 40. Ciò significa che in modalità 1 (mode 1) si hanno 40 colonne. Si prema **[ENTER]**: si otterrà come risposta un messaggio Syntax error ma, non ci si preoccupi è solo un modo veloce per porre il computer in attesa di istruzioni. Infatti esso visualizzerà il messaggio Ready.

Si scriva ora: mode 0 **[ENTER]**

Si potrà notare che i caratteri scritti sullo schermo sono molto larghi. Si prema ancora il tasto 1 e lo si mantenga premuto fino a quando non si sono riempite due linee di 1. Se si conta ora il numero di 1 presenti su una linea si vedrà che questo è pari a 20. Ciò significa che in modalità 0 (mode 0) si hanno 20 colonne. Si prema ancora **[ENTER]**.

Si scriva ora: mode 2 **[ENTER]**

Si vedrà che questa è la modalità con caratteri più piccoli, infatti se si scrive una riga piena di 1 si potrà contarne 80. Ciò significa che in modalità 2 (mode 2) vi sono 80 colonne. Quindi:

Mode 0 = 20 colonne
Mode 1 = 40 colonne
Mode 2 = 80 colonne

Si prema infine ancora una volta **[ENTER]**.

COLORI

Vi è una scelta di 27 colori. Su un monitor a fosfori verdi (GT 65) questi vengono visualizzati in scala di verdi. Se si acquistato un monitor GT 65 è possibile acquistare un Modulatore/Amplificatore MP2 AMSTRAD al fine di usare il colore del televisore.

In Mode 0, dei 27 colori disponibili è possibile visualizzarne contemporaneamente sullo schermo 16.
In Mode 1, dei 27 colori disponibili è possibile visualizzarne contemporaneamente sullo schermo 4.
In Mode 2, dei 27 colori disponibili è possibile visualizzarne contemporaneamente sullo schermo 2.

E' possibile poter modificare il colore del bordo della pagina (BORDER), dell'area in cui appaiono i caratteri (PAPER) o del carattere stesso (PEN) in modo indipendente l'uno dall'altro.

I 27 colori disponibili sono elencati nella Tabella 1 ognuno dei quali ha associato il numero di riferimento dell'inchiostro (INK).

TABELLA DEI COLORI

Numero inchiostro (INK)	Colore/Inchiostro (INK)	Numero inchiostro (INK)	Colore/Inchiostro (INK)
0	Nero	14	Blu pastello
1	Blu	15	Arancio
2	Blu brillante	16	Rosa
3	Rosso	17	Magenta pastello
4	Magenta	18	Verde brillante
5	Malva	19	Verde mare
6	Rosso brillante	20	Azzurro brillante
7	Violetto	21	Verde limone
8	Magenta brillante	22	Verde pastello
9	Verde	23	Azzurro pastello
10	Azzurro	24	Giallo brillante
11	Blu cielo	25	Giallo pastello
12	Giallo	26	Bianco brillante
13	Bianco		

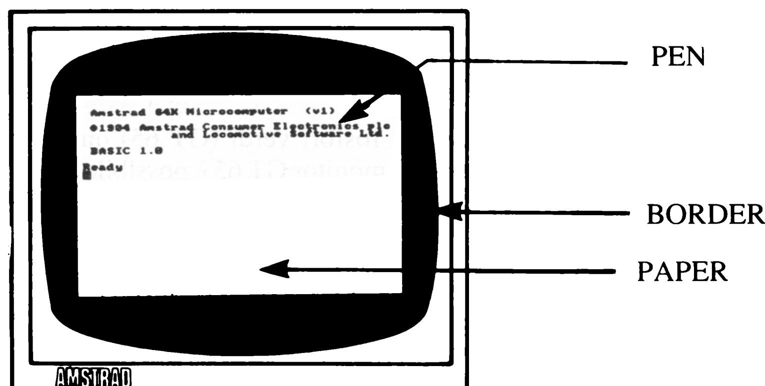
Tabella uno: numeri inchiostro e colori

Come appena spiegato quando si accende il computer questo è in Mode 1. Per riportarlo in tale modalità si scriva:

mode 1 **[ENTER]**

I colori di BORDER, PAPER e PEN che si ottengono all'accensione del computer sono:
BORDER (bordo): colore numero 1 (blu)
PAPER (area di visualizzazione dei caratteri: schermo): colore numero 1 (blu)
PEN (caratteri): colore numero 24 (giallo brillante)

Figura pagina F3.2



BORDER è il bordo che racchiude l'area di visualizzazione dei caratteri (PAPER) (si noti che all'atto dell'accensione del computer sono entrambi blu). PAPER è l'area all'interno del bordo in cui vengono visualizzati i caratteri. PEN è il colore di tali caratteri.

Per spiegare ciò in maggior dettaglio possiamo associare PEN e PAPER dello schermo all'inchiostro ed al colore di un foglio di carta. Come è possibile modificare l'inchiostro di una penna così è possibile cambiare il colore dei caratteri di uno schermo. Analogamente come è possibile modificare il colore della carta di un block notes, così è possibile modificare il colore dello schermo.
Per cambiare il colore del bordo (BORDER) si scriva:

border 0 **[ENTER]**

Si vedrà modificare il colore del bordo da blu a nero: se si fa riferimento alla Tabella 1 si potrà vedere che il numero 0 corrisponde al colore nero. Il bordo può essere modificato in uno qualsiasi dei colori elencati scrivendo: border ed il numero del colore desiderato.

Si scriva ora:

cls **[ENTER]**

per pulire lo schermo.

Per modificare il colore dello schermo (PAPER) si scriva:

paper 2 **[ENTER]**

Si vedrà il colore di fondo in fianco alla parola Ready modificato in azzurro brillante. Si scriva ora:

cls **[ENTER]**

per visualizzare lo schermo nel nuovo colore.

Per modificare il colore dei caratteri (PEN) si scriva:

PEN 3 **[ENTER]**

Si vedrà il colore dei caratteri modificati e la parola Ready stampata il rosso brillante.

Si scriva ora:

cls **[ENTER]**

Consultando la Tabella 2 sarà molto semplice comprendere cosa è stato fatto fino ad ora. Quando si è acceso il computer per la prima volta si è visto che il colore dello schermo era fissato a 0. Se si consulta la prima colonna della Tabella 2 si vedrà che il numero di schermo è 0. Se si guarda sulla stessa linea in Mode 1 si vedrà che tale colore è 1.

Se si guarda ora la tabella dei colori (Tabella 1) si vedrà che il numero 1è equivalente al colore blu che è il colore visualizzato all'accensione del computer.

Abbiamo appena modificato tale numero in 2. Si cerchi il numero 2 nella colonna a destra della Tabella 2: si potrà vedere che nella colonna mode 1 il numero di colore è 20. Si guardi ora la Tabella 1; si potrà notare che il colore 20 è azzurro brillante.

Colore dell'inchiostro

Nr. schermo/carattere	Mode 0	Mode 1	Mode 2
0	1	1	1
1	24	24	24
2	20	20	1
3	6	6	24
4	26	1	1
5	0	24	24
6	2	20	1
7	8	6	24
8	10	1	1
9	12	24	24
10	14	20	1
11	16	6	24
12	18	1	1
13	22	24	24
14	Lampeggiante 1,24	20	1
15	" 16,11	6	24

Tabella : riferimenti PAPER/PEN/MODE/INK



Il colore del carattere (PEN) usato all'atto dell'accensione del computer era 1. Consultando la Tabella 2 si potrà notare che nella colonna Mode 1 il carattere ha come colore 24. Se si consulta la Tabella 1 si potrà vedere che il numero 24 corrisponde al giallo brillante che è il colore del carattere visualizzato quando si accende il computer.

Abbiamo appena modificato PEN nel numero 3. Guardando la Tabella 2 si potrà notare che il numero 3 corrisponde al numero 6 in Mode 1. Nella Tabella 1 si vedrà, invece, che il colore 6 è rosso brillante.

Attualmente stiamo usando il numero 2 per lo schermo ed il 3 per i caratteri. Possiamo ora modificarne i colori. Per far ciò usiamo il comando INK. Tale comando necessita di due numeri: il primo è il numero equivalente al carattere (PEN) o allo schermo (PAPER) da modificare, il secondo è il colore del carattere (PEN) o dello schermo (PAPER) in cui modificarlo. Illustreremo ora, come esempio, come modificare il colore di PAPER 2 in nero e il colore di PEN 3 in bianco brillante.

Nella Tabella 1 si può vedere che il numero corrispondente al nero è 0 e il numero corrispondente al bianco brillante è 26.

Si scriva ora:

ink 2,0 **[ENTER]**

(Come appena detto, 2 è il numero di schermo attuale, e 0 è nero).

Si scriva ora:

ink 3,26 **[ENTER]**

(Come appenda detto, 3 è il numero di carattere attuale, 26 è bianco brillante).

Ora si riavvi il computer mantenendo premuti i tasti **[CTRL][SHIFT]** e **[ESC]**.

Quando il computer viene acceso o riavviato usando i tasti **[CTRL][SHIFT][ESC]**, come recentemente spiegato, il numero di schermo (PAPER) è 0 e quello dei caratteri (PEN) è 1. Il colore dello schermo (PAPER) è 1 (blu) e quello dei caratteri (PEN) è 24 (giallo brillante). Questi sarebbero stati scritti come ink 0,1 per PAPER e ink 1,4 per PEN. Per modificare i caratteri in nero su sfondo bianco si può scrivere:

ink 0,0 **[ENTER]**

E quindi:

ink 1,26 **[ENTER]**

COLORI LAMPEGGIANTI

E' possibile rendere il colore dei caratteri lampeggiante: commutando cioè da un colore ad un altro. Ciò è ottenibile mediante un ulteriore numero di colore inserito nel comando INK (carattere).

Per vedere lampeggiare i caratteri dello schermo dal colore bianco brillante al rosso brillante si provi a scrivere:

ink 1,26,6 **[ENTER]**

In questo caso, 1 è il colore del carattere (PEN) mentre 26 è il colore bianco brillante e 6 il colore alternativo, rosso brillante.

E' inoltre possibile rendere il colore dello schermo (PAPER) in fianco ai caratteri lampeggianti lampeggiante anch'esso: da un colore in un altro. Ciò può essere ottenuto aggiungendo al comando INK dello schermo un ulteriore numero.

Per vedere lampeggiare lo schermo dal colore verde al giallo brillante si può scrivere:

ink 0,9,24 **[ENTER]**

In questo caso 0 è il numero di PAPER (schermo) mentre 9 è il colore verde e 24 il colore alternativo, giallo brillante.

Si riavvi a questo punto il computer mediante i tasti **[CTRL][SHIFT][ESC]**.

Si noti che in Mode 0 due dei caratteri (numeri 14 e 15) insieme a due dello schermo (numeri 14 e 15) sono i parametri per difetto dei colori lampeggianti. In altre parole non è necessario aggiungere un ulteriore numero al comando INK.

Si scriva:

mode 0 **[ENTER]**
pen 15 **[ENTER]**

si vedrà sullo schermo la parola Ready lampeggiare nei colori blu cielo e rosa.

Si scriva ora:

paper 14 **[ENTER]**
cls **[ENTER]**

Oltre alla parola Ready lampeggiante in blu cielo e rosa si potrà vedere che anche lo schermo lampeggia: modificando alternativamente il colore da giallo a blu.

E' possibile modificare questi parametri per difetto scrivendo una nuova istruzione INK per PEN o per PAPER. Per modificare il colore di PEN in modo che lampeggi in nero e bianco brillante si scriva:

ink 15,0,26 **[ENTER]**

In questo caso 15 è il numero di carattere (PEN) mentre 0 è il colore nero e 26 il colore alternativo ovvero bianco brillante.

E' possibile, infine rendere anche il bordo lampeggiante: per fare ciò occorre aggiungere nel comando BORDER un ulteriore numero di colore. Si scriva:

border 6,9 **[ENTER]**

Si vedrà il bordo lampeggiare nei colori rosso brillante e verde.

Si riavvi a questo punto il computer mantenendo premuti i tasti **[CTRL][SHIFT][ESC]**.

PROGRAMMA DIMOSTRATIVO

Per una ulteriore dimostrazione dei colori disponibili si scriva il seguente programma e lo si esegua (mediante il comando run).

Abbiamo incluso nel programma dei suoni: tale caratteristica verrà spiegata successivamente.

```
10 mode 0: ink 0,2: ink 1,24: paper 0 [ENTER]
20 pen 1: for b=0 to 26: border b [ENTER]
30 locate 3,12: print "COLORE DEL BORDO";b [ENTER]
40 sound 4,(40-b) [ENTER]
50 for t=1 to 600:next t:next b:cls[ENTER]
60 for p=0 to 15:paper p:pen 5:print "schermo";p :print [ENTER]
70 for n=0 to 15: pen n:print "carattere";n[ENTER]
80 sound 1, (n*20+p)[ENTER]
90 for t=1 to 100:next t:next n[ENTER]
100 for t=1 to 1000:next t:cls:next p[ENTER]
110 cls: paper 0:pen 1: locate 7,12:print "FINE": for t=1 to 2000: next t[ENTER]
120 mode 1:border 1:ink 0,1:ink 1,24: paper 0: pen 1[ENTER]
```

run **[ENTER]** .

GRAFICA

Da questo punto in poi, non verrà richiesto tutte le volte di premere **[ENTER]**; si assumerà infatti che questo venga fatto automaticamente dall'utente.

Vi sono diversi simboli memorizzati in memoria: per visualizzarne uno si usa la parola chiave chr$(). Nelle parentesi tonde deve essere inserito un numero compreso tra 32 e 255.

Si riavvi il computer mantenendo premuti i tasti **[CTRL][SHIFT][ESC]** e successivamente si scriva:

```
print chr$(250)
```

Verrà visualizzato sullo schermo il carattere corrispondente al numero 250 che è un uomo che cammina verso destra.

Per vedere sullo schermo tutti i caratteri associati ai numeri si scriva il seguente programma ricordandosi di premere **[ENTER]** al termine di ogni linea.

```
10 for n=32 to 255: print n;chr$(n);
20 next n
run
```

L'elenco dei caratteri corrispondenti ai numeri ed i loro simboli è riportato nell'Appendice III di questo manuale.

LOCATE

Questo comando è usato per posizionare il cursore in un punto specifico dello schermo. A meno che non sia stato modificato da un comando locate, il cursore viene posizionato nell'angolo in alto a sinistra dello schermo che corrisponde alle coordinate x,y = 1,1 (x la posizione orizzontale e y la verticale). In Mode 1 vi sono 40 colonne e 25 linee. Per posizionare, in Mode 1, il cursore al centro della prima lineaa si devono usare le coordinate 20,1.

Si scriva (ricordando di premere **[ENTER]** al termine di ogni linea):

mode 1 .... *lo schermo viene pulito, ed il cursore posizionato in alto a sinistra*

```
10 locate 20,1
20 print chr$(250)
run
```

Per provare che il cursore si trova effettivamente sulla prima linea si scriva:

```
border 0
```

Il bordo sarà ora nero e si vedrà il simbolo dell'uomo in mezzo alla prima linea.

In Mode 0 vi sono solo 20 colonne ma 25 linee. Se si scrive ora:

```
mode 0
run
```

si vedrà che il simbolo dell'uomo appare nell'angolo in alto a destra dello schermo. Questo perchè in modalità 0 la coordinata 20 corrisponde all'ultima colonna.

In Mode 2 vi sono 80 colonne e 25 linee. Usando lo stesso programma si potrà molto probabilmente immaginare dove apparirà ora l'uomo. Si scriva:

```
mode 2
run
```

Si ritorni ora in modo Mode 1 scrivendo:

```
mode 1
```

Si provi a questo punto a fare degli esperimenti modificando i comandi locate e chr$() e variando la posizione dei caratteri nello schermo. A titolo di esempio si provi a scrivere:

```
locate 20,12: print chr$(240)
```

Si vedrà una freccia in mezzo allo schermo. Si noti che in questa istruzione:

20 è la coordinata (x) orizzontale (nell'area 1- 40)
12 è la coordinata (y) verticale (nell'area 1 - 25)
240 è il numero corrispondente al simbolo (da 32 a 255)

Per visualizzare il simbolo corrispondente al numero 250 in tutto lo schermo si scriva il seguente programma:

```
5 cls
10 for x = 1 to 39
20 locate x,20
40 print chr$(250)
50 next x
60 goto 5
run
```

Per fermare il programma si prema due volte il tasto **[ESC]**

Per eliminare il carattere precedente prima di visualizzarne un altro si scriva:

```
40 print "     "; chr$(250)
```

(Tale linea sostituisce automaticamente la linea 40 precedentemente scritta).

Si scriva ora:

```
run
```

Per migliorare il movimento del carattere nello schermo si aggiunga la seguente linea:

```
30 call &bd19
```

Questo programma può essere ulteriormente migliorato aggiungendo alcuni ritardi ed usando diversi simboli.

Si scriva:

```
list
```

Si aggiungano ora le seguenti linee di programma:

```
60 for n = 1 to 300 : next n
65 for x = 39 to 1 step-1
70 locate x,20
75 call &bd19
80 print chr$(251);"    "
85 next x
90 for n = 1 to 300:next n
95 goto 10
run
```

Si provi ora questo interessante programma: abbiamo aggiunto alcuni comandi che verranno spiegati nei capitoli seguenti. Per il momento si scriva:

```
new

10 mode 1
20 locate 21,14:print chr$(244)
30 tag
40 for x=0 to 624 step 2
50 mover -16,0
60 if x<308 or x>340 then y=196:goto 90
70 if x<324 then y=x-104:goto 85
80 y=536-x
85 sound 1,0,20,7
90 move ox, oy:print "    ";:ox=x:oy=y
100 move x,y
110 if (x mod 4) = 0 then print chr$(250); else print chr$(251);
120 for n=1 to 4: call &bd19:next n
130 next x
140 tagoff
150 goto 20
run
```

PLOT

Diversamente dal comando locate, plot è usato per determinare la posizione del cursore grafico usando le coordinate pixel (abbreviazione di picture element). Un pixel è un segmento molto piccolo sullo schermo.

Si noti che il cursore grafico non è visibile ed è diverso dal cursore dei caratteri.

Vi sono 640 pixel orizzontali e 400 verticali. Le coordinate x,y sono posizionate a partire dall'angolo in basso a sinistra dello schermo: tale punto ha coordinate 0,0. Contrariamente al comando locate le coordinate non differiscono nelle varie modalità (Mode 0, 1 o 2).

Per vedere ciò si riavvi il computer mantenendo premuti i tasti **[CTRL][SHIFT][ESC]** e si scriva (premendo sempre **[ENTER]**):

```
plot 320,200
```

In mezzo allo schermo verrà visualizzato un piccolo punto.

Si modifichi ora il Mode scrivendo:

```
mode 0
plot 320,200
```

Il punto sarà posizionato ancora in mezzo allo schermo ma è un po' più grande. Si modifichi ancora la modalità e si scriva lo stesso comando per vederne l'effetto in Mode 2. Si scriva:

```
mode 2
plot 320,200
```

Il punto è ancora in centro ma ancora più grande.

Per potersi rendere conto di persona dell'uso di tale comando si provi a posizionare diversi punti nello schermo nelle varie modalità. Al termine si ritorni in Mode 1 e si pulisca lo schermo scrivendo:

```
mode 1
```

DRAW

Si riavvi a questo punto il computer mantenendo premuti i tasti **[CTRL][SHIFT][ESC]**. Il comando DRAW disegna una linea dalla posizione corrente del cursore grafico. Per vedere in dettaglio tale comando disegnamo un rettangolo sullo schermo usando il seguente programma. Iniziamo a far ciò posizionando il cursore grafico mediante un comando PLOT. Disegnamo poi una linea partendo dalla posizione del cursore e verso l'angolo in alto a sinista, poi da questo punto verso l'angolo destro, ecc.

Si scriva:

```
5 cls
10 plot 10,10
20 draw 10,390
30 draw 630,390
40 draw 630,10
50 draw 10,10
60 goto 10
run
```

Si prema due volte **[ESC]** per fermare il programma.

Si aggiungano ora al programma le seguenti linee: si disegnerà all'interno del primo rettangolo un secondo. Si scriva:

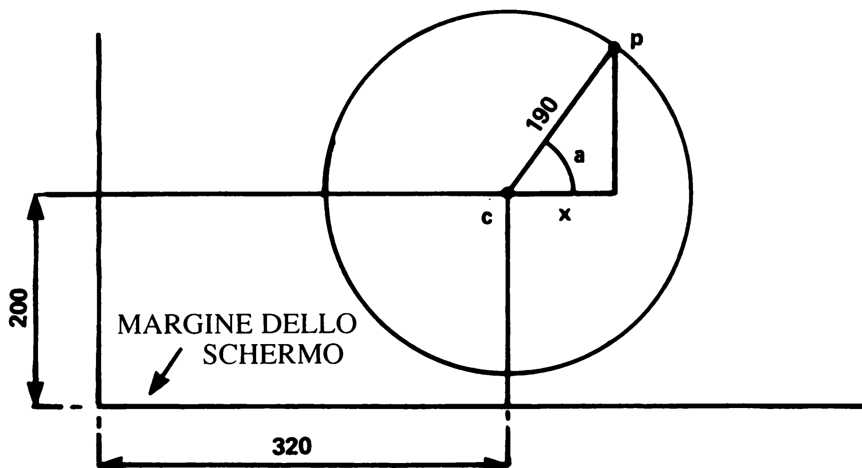
```
60 plot 20,20
70 draw 20,380
80 draw 620,380
90 draw 620,20
100 draw 20,20
200 goto 10
run
```

Si prema due volte **[ESC]** per fermare il programma.

CERCHI

I cerchi possono essere sia disegnati punto per punto che tutti in una volta. Un metodo per formare il cerchio consiste nel disegnare ogni punto x,y della circonferenza. Facendo riferimento alla figura sotto si può notare che il punto p della circonferenza può essere disegnato usando le coordinate x ed y. Queste sono:

x= 190*cos(a) e y = 190*sin(a)



NEW

Abbiamo scritto prima del programma stesso la parola chiave new. Ciò indica al computer di pulire la memoria come si fa in modo analogo quando lo si riavvia (**[CTRL][SHIFT][ESC]**). Tale comando, comunque, 'svuota' solo la memoria e non lo schermo. Ciò è utile quando si vuole mantenere visibile sullo schermo un vecchio programma per tenerlo come riferimento per uno nuovo.

Disegniamo i punti di un cerchio.

Nel programma precedente abbiamo disegnato i punti del cerchio rispetto all'angolo in basso a sinistra dello schermo. Se ora vogliamo posizionare il cerchio in mezzo allo schermo dobbiamo posizionare il centro del cerchio alle coordinate 320, 200 e successivamente disegnare tutti i punti del cerchio rispetto a tale posizione.

Un tale programma potrebbe essere come questo che segue:

```
new
5 cls
10 for a=1 to 360
15 deg
20 plot 320,200
30 plot 320 + 190*cos(a),200+190* sin(a)
40 next
run
```

Il raggio del cerchio può essere ridotto diminuendo il valore 190 (ovvero il numero di pixel).

Per vedere un diverso modo di tracciamento del cerchio(in radianti) si cancella la linea 15 del programma scrivendo:

```
15
```

Per vedere disegnato un cerchio mediante linee che partono dal centro si editi la linea 30 e si sostituisca la parola plot con la parola draw. Tale linea dovrà essere come segue:

```
draw 320+190*cos(a), 200+190*sin(a)
```

Si provi ad apportare tale modifica e ad eliminare ancora la linea 15.

Si noterà che la linea 40 del programma next e non next a.

Nei programmi è permesso scrivere semplicemente next. Il computer elaborerà quella espressione for a cui associato è next. Nei programmi in cui vi sono diversi cicli for e next si potrebbe voler aggiungere il nome appropriato dopo la parola next al fine di identificare il comando quando si studia il programma.

ORIGIN

Nel programma precedente abbiamo usato il comando plot per posizionare il centro di un cerchio e per aggiungervi le coordinate x,y. Invece di aggiungere tali coordinate al punto centrale disegnato, è possibile usare il comando origin. Esso posizionerà il centro del cerchio e le coordinate di tutti i punti della circonferenza (con passo equivalente ad 1 grado) dall'origine. Si provi a scrivere:

```
new
5 cls
10 for a = 1 to 360
15 deg
20 origin 320,200
30 plot 190*cos(a),190*sin(a)
40 next
run
```

E' possibile anche in questo caso modificare la linea 15 o la 30 togliendo deg e scrivendo draw per disegnare il cerchio partendo dal centro.

```
new
5 cls
10 for a = 1 to 360
15 deg
20 origin 196,282
30 plot 50*cos(a),50*sin(a)
40 origin 442,282
50 plot 50*cos(a),50*sin(a)
60 origin 196,116
70 plot 50*cos(a),50*sin(a)
80 origin 442,116
90 plot 50*cos(a),50*sin(a)
100 next
run
```

Anche in questo caso è possibile eliminare la linea 15 e modificare le 30,50,70 e 90 usando il comando draw.

GOSUB RETURN

Se all'interno del programma vi sono delle istruzioni che devono essere ripetute diverse volte, queste possono essere scritte come una sotto-routine e richiamate con un comando gosub a cui fa seguito un numero.

La fine di una routine viene contraddistinta da una istruzione return: in tal punto il computer tornerà ad eseguire la linea seguente l'istruzione gosub.

Nel programma precedente l'istruzione plot 50*cos(a),50*sin(a) è stata ripetuta quattro volte. Questa istruzione può essere scritta come una sotto-routine e richiamata in azione mediante il comando gosub . Si scriva:

```
new
5 cls
10 for a = 1 to 360
15 deg
20 origin 196,282
30 gosub 120
40 origin 442,282
50 gosub 120
60 origin 196,116
70 gosub 120
80 origin 442,116
90 gosub 120
100 next
110 end
120 plot 50*cos(a),50*sin(a)
130 return
run
```

Si noti che l'istruzione end è stata usata nella linea 110; se non fosse stato così il programma non si sarebbe fermato alla istruzione 100 ma avrebbe ripetuto ancora una volta l'istruzione 120 la quale invece deve essere richiamata solo con una istruzione gosub.

Per concludere questa parte si provi a scrivere il seguente programma che include numerosi comandi e parole chiave ormai note. Si scriva:

```
new
10 mode 0:border 6:paper 0:ink 0,0
20 gosub 160:for x=1 to 19:locate x,3
30 pen 15:print" ";chr$(238)
40 for t=1 to 50:next t:sound 2,(x+100)
50 next x:gosub 160:for b=3 to 22
60 locate 20,b:pen 7:print chr$(252)
70 cls:gosub 160:next b
80 sound 2,0,100,15,0,0,1
90 gosub 160:border 16,24:locate 20,25
100 pen 14:print chr$(253);
110 for t =1 to 1000:next t
120 border 6:gosub 160: for f=3 to 24
130 locate 10,(25-f):pen 2
140 print chr$(144):cls:gosub 160
150 sound 7,(100-f),5:next f:goto 10
160 locate 10,25:pen 12
170 print chr$(239):return
run
```

SUONO

Gli effetti sonori sono generati da un altoparlante incorporato nel computer. Se si sta usando un modulatore/alimentatore MP2 e un televisore domestico si porti il volume di quest'ultimo al minimo.

Il livello del suono può essere regolato per mezzo del controllo di VOLUME posto sulla destra del computer. Il suono può inoltre essere inviato ad una presa di ingresso ausiliare del proprio stereo, usando la presa (I/O) situata sulla sinistra sul pannello posteriore. Ciò permetterà di sentire il suono emesso attraverso le casse acustiche del proprio stereo o mediante una cuffia.

Il comando SOUND ha sette parametri. I primi due devono essere specificati gli altri sono opzionali. Il comando deve essere scritto come:

SOUND stato del canale, periodo della nota, durata, volume, inviluppo del volume, inviluppo del tono, periodo di rumore.

Nei seguenti esempi scriveremo 1 come stato del canale;

PERIODO DELLA NOTA

Facendo riferimento alla Appendice VII si vedrà che la nota DO centrale ha un periodo di 478. Si scriva:

```
new
10 sound 1,478
run
```

Si udrà una breve nota corrispondente al DO centrale della durata di 0,2 secondi.

DURATA

Se nel comando sound non si specifica la durata questa sarà di 0,2 secondi. L'unità di tempo è equivalente a 0,01 secondi. Per far durare una nota 1 secondo si deve usare 100; per 2 secondi si deve usare 200. Si scriva:

```
10 sound 1,478,200
run
```

Si udrà la nota Do centrale per la durata di 2 secondi.

VOLUME

Questo numero specifica il volume iniziale della nota. Il numero può essere compreso tra 0 e 7. Se comunque è specificato un inviluppo del volume tale area può essere estesa tra 0 e 15. Se non si specifica alcun numero viene assunto per difetto 4. Si scriva:

```
10 sound 1,478,200,3
run
```

Si noti il volume di tale suono. Lo si riscriva ora aumentando il volume:

```
10 sound 1,478,200,7
run
```

Si potrà notare che tale volume è molto più alto.

INVILUPPO DEL VOLUME

Il comando di inviluppo del volume è env. Questo ha normalmente 4 parametri: gli ultimi 3 possono apparire in una delle 5 sezioni disponibili. Useremo solo uno di questi. Ulteriori spiegazioni si trovano nel capitolo 6.

env numero di inviluppo, numero di passi, ampiezza (dimensione) del passo, tempo del passo.

NUMERO DELL'INVILUPPO

Questo numero fornisce un particolare inviluppo e può essere quindi specificato nel comando sound. La gamma di tale numeri è compresa tra 0 e 15.

NUMERO DI PASSI

Viene usato insieme al tempo del passo. Si potrebbe, ad esempio, avere 10 passi per ogni secondo. In tal caso il numero di passi è 10. La gamma di tali numeri è compresa tra 0 e 127.

AMPIEZZA DEI PASSI

Ogni passo può variare in ampiezza da un livello 0 a 15 tenendo conto dell'ultimo passo. Il volume di livello 15 è equivalente a quello del comando sound. Il passo, comunque, può essere regolato tra -128 e +127 quindi non si può solo variare l'ampiezza in alto o in basso in modo ovvio ma, la si può variare usando valori maggiori di 15 per ottenere effetti strani. La gamma di ampiezza di tali numeri varia da -128 a +127.

TEMPO DEL PASSO

Questo numero specifica l'unità di tempo intercorrente tra due passi in 0,001 secondi (1/100esimo di secondo). La gamma di tale passo varia da 0 a 256. Il tempo più lungo tra due passi è quindi di 2.56 secondi.

Si provi a scrivere il seguente programma:

```
5 env 1,10,1,100
10 sound 1,284,1000,1,1
run
```

La linea 10 specifica un suono con periodo della nota di 284, di durata 10 secondi con volume iniziale di 1, un inviluppo di volume pari a 1 ogni secondo (100x0,01 secondi).

Si modifichi la linea 5 in ognuno dei seguenti modi e si esegua ogni volta il programma per sentire l'effetto di modifica dell'inviluppo.

```
5 env 1,100,1,10
5 env 1,100,2,10
5 env 1,100,4,10
5 env 1,50,20,20
5 env 1,50,2,20
5 env 1,50,15,30
```

Ed infine si provi la seguente:

```
5 env 1,50,2,10
```

Si sarà notato che dimezzando il suono il livello resta costante. Questo perchè il numero di passi era 50 ed il tempo per ogni passo pari a 0,1 secondo. Del resto, la lunghezza dell'ampiezza variava solo di 5 secondi mentre la durata del suono nel comando sound nella linea 10 era pari a 10 secondi (numero 1000).

Si provi ad effettuare delle prove per vedere il tipo di suono che si riesce a creare.

INVILUPPO DELLA NOTA

Il comando di inviluppo della nota è ent.

Questo ha normalmente 4 parametri.

Gli ultimi 3 possono apparire in una delle 5 sezioni disponibili. Useremo solo uno di questi. Ulteriori spiegazioni sono disponibili nel capitolo 6.

ent numero di inviluppo, numero di passi, ampiezza della nota, tempo del passo.

NUMERO DELL'INVILUPPO

Questo numero fornisce un particolare inviluppo: può essere quindi specificato nel comando sound. La gamma di tale numeri compresa tra 1 e 15.

NUMERO DI PASSI

Viene usato insieme al tempo del passo. Si potrebbe, ad esempio, avere 10 passi per ogni secondo. La gamma di tali numeri compresa tra 0 e 239.

AMPIEZZA DELLA NOTA

L'ampiezza della nota ad ogni passo può essere regolata tra -128 e +127. I passi negativi incrementano la frequenza della nota (rendono la nota più alta). L'ampiezza minore equivale a 0: questo deve essere ricordato quando si calcola l'inviluppo di tono. La gamma completa del periodo del tono è riportata nell'Appendice VII. La gamma di ampiezza di tali numeri varia da -128 a +127.

TEMPO DEL PASSO

Questo numero specifica l'unità di tempo intercorrente tra due passi in 0,001 secondi (1/100esimo di secondo). La gamma di tale passo varia da 0 a 255. Il tempo più lungo tra i passi è quindi di 2.55 secondi.

Si provi a scrivere il seguente programma:

```
5 ent 1,100,2,2
10 sound 1,284,200,7,0,1
run
```

La linea 10 specifica un suono con periodo della nota di 284, di durata 10 secondi con volume iniziale di 7, senza inviluppo di volume (rappresentato da 0) e con inviluppo della nota pari a 1.

La linea 5 ha numero di inviluppo della nota uguale a 1 formato da 100 passi, con incremento del periodo della nota (riduzione della frequenza) di 2 ogni 0,02 secondi (2/100esimi di secondo).

Si modifichi la linea 5 in ognuno dei seguenti modi e si esegua ogni volta il programma per sentire l'effetto di modifica dell'inviluppo.

```
5 ent 1,100,-2,2
5 ent 1,10,4,20
5 ent 1,10,-4,20
```

Ora si sostituisca il comando sound e l'inviluppo della nota scrivendo:

```
5 ent 1,2,17,70
10 sound 1,142,140,15,0,1
15 goto 5
run
```

Si prema il tasto **[ESC]** due volte per fermare il programma.

Ora è possibile porre in un comando sound l'inviluppo del volume, e l'inviluppo della nota per creare diversi suoni. Si scriva:

```
new
5 env 1,100,1,3
10 ent 1,100,5,3
20 sound 1,284,300,1,1,1
run
```

Si sostituisca a questo punto la linea 10 scrivendo:

```
10 ent 1,100,-2,3
```

Si sostituiscano ora tutte le linee scrivendo:

```
5 env 1,100,2,2
10 ent 1,100,-2,2
20 sound 1,284,200,1,1,1
```

Si provi ad effettuare delle variazioni.

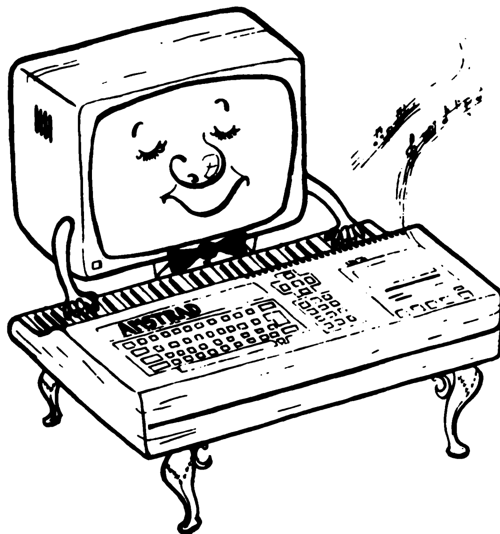
RUMORE

Alla fine del comando sound è possibile inserire un rumore. La gamma di rumori disponibili è compresa tra 1 e 31. Si provi ad aggiungere alla fine del comando sound il numero di rumore usando ancora il comando env.

Si sostituisca la linea 5 e la 20 con le seguenti:

```
5 env 1,100,3,1
20 sound 1,200,100,1,1,1,5
run.
```

Si provi a creare dei suoni insoliti modificando l'inviluppo del volume e il comando sound, con o senza il rumore.



1. Avvio:

Se coloro che hanno saltato l'introduzione per i principianti (che comprende i particolari riguardanti l'accensione e l'uso della tastiera) dovessero trovare la terminologia poco chiara, potranno tornare ad esaminare la parte introduttiva del corso .

Argomenti riportati in questo capitolo:

* Convenzioni usate in questo manuale
* Accensione
* Familiarizzazione con la tastiera

Non importa quanto si conoscano la programmazione ed i computer, si prega di seguire queste brevi istruzioni di installazione. Se si ha già aperto la confezione, e si è impazienti di iniziare, questo capitolo riporta tutto quanto è necessario sapere per soddisfare una comprensibile curiosità iniziale e per iniziare ad utilizzare il BASIC. Questa parte è dedicata a coloro che hanno già qualche conoscenza di base nel campo dei computer. I principianti dovrebbero iniziare dal capitolo introduttivo.

IMPORTANTE - DEVE essere letto:

Terminologia.

Per rendere più chiare le differenze nei riferimenti tra i tasti della tastiera, ed i testi del listato del programma, da ora in poi verranno usate le seguenti convenzioni:

[ENTER] : Vengono visualizzati in questo modo (tra parentesi quadre) i tasti che non corrispondono ad un carattere nella stampa su video.

QWERTYUIOP : Vengono visualizzati in questo modo (senza parentesi quadre) i tasti che corrispondono alla stampa di un carattere.

10 FOR n = 1 TO 1000 : Vengono visualizzati in questo modo i testi che appaiono allo schermo o che devono essere inseriti da tastiera. Si noti la differenza tra lo zero (0) e la lettera maiuscola O.

Si assumerà che ogni linea di programma o di comando venga terminata con **[ENTER]**, e ciò non verrà ripetuto nel prosieguo del manuale.

Inoltre si assumerà che, dopo aver terminato un programma, per la sua esecuzione si dovrà inserire run. Il BASIC converte automaticamente in MAIUSCOLO ogni parola chiave immessa in minuscolo; ciò sarà visibile usando il comando LIST. Gli esempi riportati d'ora in avanti useranno lettere MAIUSCOLE, in quanto questo è l'aspetto che il programma assume quando viene LISTato. Se si immetteranno le parole chiave in minuscolo sarà anche più facile trovare gli errori di battitura in quanto le parole chiave non corrette appariranno in minuscolo.

1.1.1 APRIAMO LA SCATOLA!

Il monitor a colori

IMPORTANTE

Si faccia riferimento alle istruzioni di installazione all'inizio di questo manuale dove vengono descritti i cavi di alimentazione.

Quando il computer sarà connesso come descritto in Fondamenti 1, si accenda il monitor e quindi il computer, usando l'interruttore che si trova sul lato destro dell'apparecchio. Dopo circa 30 secondi il monitor visualizzerà:

```
Amstrad 64K Microcomputer  (v1)
©1984 Amstrad Consumer Electronics plc
          and Locomotive Software Ltd.
BASIC 1.0
Ready
```

Questo è conosciuto come messaggio di "reset" ed indica che il computer è stato reinizializzato - una condizione che avviene al momento dell'accensione e dopo un reset da tastiera prodotto dalla combinazione dei tre tasti **[CTRL][SHIFT]** e **[ESC]** premuti in sequenza e mantenuti premuti contemporaneamente - si provi questa operazione ora, prima di fare qualunque altra cosa.

Si regoli il comando di luminosità BRIGHTNESS sul lato destro del monitor. I colori al momento dell'accensione sono fissati e se si desidera cambiarli (lettere color oro su sfondo blu) occorre inserire alcune istruzioni e sarà necessario passare all'introduzione alla grafica e alle parole chiave del BASIC nel Capitolo 8. Se non si intende saltare avanti, ecco un breve programma che fornisce una delle combinazioni di colori migliori e più leggibili per l'immissione di testi, usando 80 colonne della modalità ad alta risoluzione.

Si immetta:

```
10 MODE 2
20 INK 1,0
30 INK 0,13
40 BORDER 13
```

Se questo programma è poco chiaro o non si riesce ad inserirlo correttamente sarà necessario ritornare al *CORSO INTRODUTTIVO PER PRINCIPIANTI* all'inizio del manuale. Si troverà che la modalità a 80 colonne di testo è probabilmente la modalità di schermo più utile per lo sviluppo di programmi - si potrebbe voler salvare questo breve programma (dopo aver letto il Capitolo 2) all'inizio di una cassetta vuota, per evitare di doverlo inserire ad ogni accensione.

ATTENZIONE! A causa dell'alta luminosità del monitor a colori, esso potrebbe provocare dolori agli occhi, se ci si pone troppo vicini ad esso o se si mantiene un livello di luminosità superiore a quello necessario per le condizioni di illuminazione ambientale. Se si dovessero sentire dolori agli occhi, si smetta di utilizzare il computer e si faccia qualcos'altro; tuttavia per evitare tali problemi:

1. Si lavori sempre con luce sufficiente per permettere di leggere con facilità il manuale. In nessun caso si legga il manuale utilizzando come illuminazione la luminosità dello schermo!

2. Si usi la minor luminosità possibile che permetta di vedere con facilità ciò che si sta facendo.

3. Ci si sieda il più possibile lontani dallo schermo.

Una lampada da tavolo posta in fianco al monitor aiuterà ad evitare i dolori agli occhi - sempre che sia posta in fianco allo schermo per evitare riflessi.

1.1.2 Monitor a fosfori verdi

Il monitor GT65 è dotato, sotto allo schermo, di tre controlli. Essi servono per permettere la regolazione della luminosità, del contrasto (la differenza tra le zone più chiare e quelle più scure dello schermo) e della linearità verticale, che permette di fissare l'immagine sullo schermo ed evitare che l'immagine ruoti sullo schermo.

Non sarà necessario regolare spesso la linearità verticale - una volta sistemata può essere dimenticata. I comandi di luminosità e contrasto potrebbero invece dover essere regolati di tanto in tanto a seconda delle condizioni di illuminazione del locale in cui il CPC 464 è posto.

Usando un monitor monocromatico (uno schermo ad un solo colore che varia l'intensità dei caratteri per permettere un contrasto tra i vari elementi dell'immagine), il messaggio che apparirà inizialmente sullo schermo sarà uguale a quello di uno schermo a colori (si veda la pagina precedente) ma il testo verrà visualizzato in verde chiaro su uno sfondo verde scuro.

Un uso troppo prolungato del computer e del monitor GT65 può produrre un dolore agli occhi, ma l'immagine più "delicata" prodotta dal monitor monocromatico risulta a lungo termine meno dannosa. In particolare sarà possibile visualizzare meglio l'immagine della modalità "80 colonne di testo" (80 lettere o numeri nella stessa linea di schermo) dal momento che la risoluzione (la capacità di visualizzare un certo numero di piccoli punti sullo schermo uno vicino all'altro senza che essi vengano "uniti") di un monitor monocromatico è intrinsecamente superiore anche a quella di un costoso monitor a colori.

Si regoli la luminosità in modo che lo schermo sia sufficientemente luminoso e che i punti che compongono i caratteri non siano troppo "impastati" tra di loro.

Per predisporre la modalità ad 80 colonne, si inserisca il breve programma che segue. Sarà possibile scegliere la modalità desiderata:

```
10 REM Formato schermo
20 FOR n=0 TO 26
30 MODE 2
40 INK 1,n
50 INK 0,(26-n)
55 BORDER n
60 LOCATE 15,12:  PRINT "Si prema un tasto stampabile qualunque per
cambiare il formato"
70 a$=INKEY$
80 IF a$="" GOTO 80
90 NEXT
100 GOTO 20
```

Questo programma illustra un altro concetto riguardante la rappresentazione dei programmi in questo manuale. Alcune linee dovranno andare a capo (poichè superano il margine destro dello schermo) e facciamo notare che nella presentazione di un programma gli spazi aggiuntivi inseriti nel punto in cui la linea si spezza non devono essere inseriti nel programma ma sono presenti solo per facilitare la lettura del listato.

Questo programa non illustra tutti i livelli di colore ma darà un'idea delle possibilità disponibili. Quando si troverà la giusta combinazione, si prema **[ESC]** due volte (apparirà il messaggio *Break*).

Da ora in poi verranno fatti molti riferimenti che riguarderanno specificamente il monitor a colori. I programmi che visualizzano effetti di colore e grafica potrebbero risultare praticamente invisibili sullo schermo monocromatico, ma è stata posta una cura particolare nella produzione di gamme di colore corrispondenti a livelli di grigio (questo argomento verrà approfondito nel Capitolo 5).

Il vantaggio nel possedere un monitor monocromatico consiste nell'immagine più netta e meno affaticante per la vista durante lo sviluppo di programmi; ma quando e se si verrà coinvolti dall'affascinante mondo dei calcolatori, la scelta del colore sarà probabilmente inevitabile.

1.1.3 Il modulatore TV / Alimentatore MP2

L'MP2 è uno degli oggetti che si vorrà probabilmente acquistare se attualmente si sta usando il CPC464 con il monitor verde GT65. Esso permette di usare il computer con l'apparecchio televisivo di casa e permette di visualizzare tutti i colori del CPC464.

> **IMPORTANTE**
>
> Per una descrizione dei cavi di collegamento e di alimentazione e delle prese e spine, si faccia riferimento alle istruzioni di installazione all'inizio di questo manuale.

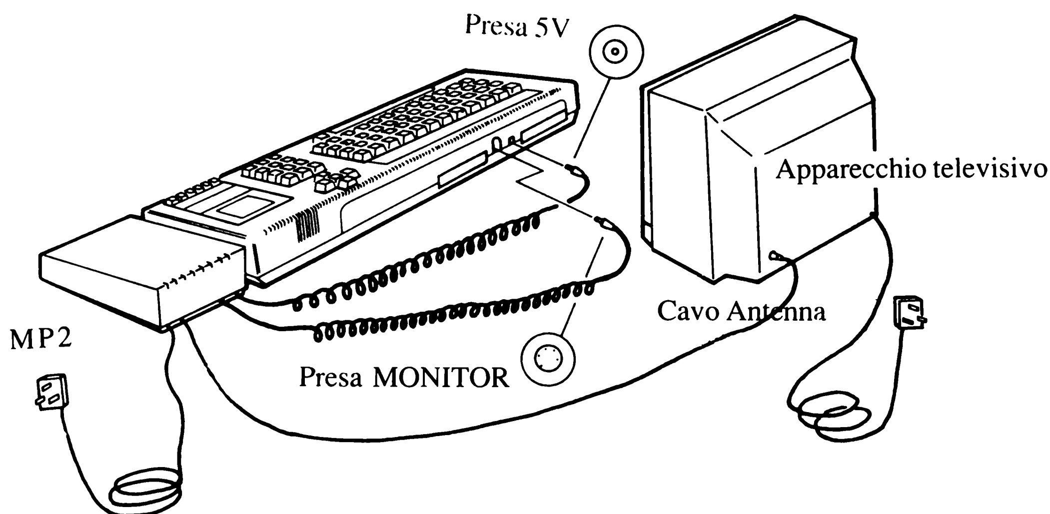


Figura 2: Connessioni tra MP2, computer e ingresso di antenna del TV

L'alimentatore/modulatore MP2 dovrebbe essere posto alla destra del computer su un tavolo vicino al TV e ad una presa di corrente, come mostrato in figura 2.

Ora si riduca il volume del TV al minimo - il CPC464 ha il proprio altoparlante interno e perciò il soffio proveniente dal TV quando il volume non è al minimo ed il computer acceso, è perfettamente normale. Si accenda il TV e poi il computer utilizzando l'interruttore POWER che si trova sul lato destro.

La spia di accensione verso il centro della tastiera dovrebbe illuminarsi e sarà così possibile sintonizzare il TV sul segnale del computer.

Se si possiede un TV con sintonia a tasti, si usi un tasto non utilizzato da altri canali. Si regolino poi i controlli di sintonia secondo le istruzioni fornite dal fabbricante (se il TV ha una scala graduata, il segnale si troverà vicino al canale 36) e si cerchi l'immagine presentata nella prossima pagina.

```
Amstrad 64K Microcomputer  (v1)

©1984 Amstrad Consumer Electronics plc
          and Locomotive Software Ltd.

BASIC 1.0

Ready
█
```

Si sintonizzi accuratamente il TV in modo da ottenere l'immagine più nitida possibile. I caratteri verranno scritti in giallo oro su sfondo blu, sebbene tali colori potrebbero variare a seconda della regolazione del proprio TV.

Se si possiede un TV con sintonia a manopola, la si ruoti finchè l'immagine qui sopra sarà perfettamente ferma (sempre vicino al canale 36).

Il segnale presentato sul video è stato modulato e poi demodulato dal televisore e dunque sarà possibile notare un certo scadimento dell'immagine. Il risultato non potrà essere della stessa qualità di quello del monitor a colori e, a seconda della qualità del televisore, si potrà trovare che la modalità di testo ad 80 colonne (modo 2) non produce risultati perfettamente nitidi e sarà forse più conveniente utilizzare il modo 1.

1.2 Primi passi

Ora si è finalmente collegato tutto: l'alimentatore dovrebbe essere collegato, il cavo video dovrebbe essere collegato ad un monitor o al modulatore ed il computer dovrebbe essere acceso in attesa di un input.

Il messaggio presentato ora sullo schermo è l'unico testo "interno" della macchina che è possibile visualizzare senza immettere alcuna istruzione dalla tastiera.

Se si conoscono già i concetti della programmazione BASIC, è possibile si sia già inserito qualche breve programma allo scopo di "familiarizzare". Si troverà il BASIC AMSTRAD familiare sotto molti aspetti e, per permettere di proseguire, mostreremo un breve programma che si potrà immettere e che mostra tutti i caratteri visualizzabili dal computer. Questo è il set di caratteri, termine usato per indicare tutte le cifre, i segni ed altri elementi stampabili che possono essere richiamati utilizzando la tastiera.

Alcuni dei caratteri che si vedranno non sono direttamente accessibili da tastiera ma solo usando il comando PRINT CHR$(<numero>) che verrà descritto più avanti in questo capitolo.

Ciò avviene perchè ogni elemento memorizzato viene posto in un elemento di memoria noto come "byte" - e come si vedrà nella Appendice II ciò significa 256 possibili combinazioni diverse. Dal momento che un computer deve usare almeno un byte per ogni carattere memorizzato (che lo si voglia o no è l'unità di memoria più piccola utilizzabile dal CPC464), dovremo utilizzare tutte le 256 possibili combinazioni, invece delle 96 che normalmente vengono utilizzate dalla maggior parte delle macchine per scrivere e gettare via le rimanenti 160 possibilità.

L'insieme dei caratteri "standard" forma un "sotto-insieme" ed è noto nel mondo dei computer come sistema ASCII, un termine derivato da:

American
Standard
Code for
Information
Interchange

L'Appendice III elenca tutti i caratteri ASCII oltre ai caratteri addizionali disponibili sul CPC464 e i relativi codici numerici.

Alcuni degli altri caratteri "non stampabili" possono essere visualizzati utilizzando combinazioni del tasto CONTROL (riportato sulla tastiera con **[CTRL]**) e di altri tasti - ma non ci si preoccupi ancora di questi problemi, prima di comprendere il funzionamento del tasto Control sarà possibile fare degli esperimenti a caso.

1.2.2

Per vedere effettivamente quali sono questi caratteri, si immetta il programma che seguirà; intendiamo risvegliare sia la curiosità dell'utente che il CPC464. Questo programma aiuterà anche a prendere confidenza con la semplicità della programmazione ed a rassicurare che se è apparso il messaggio iniziale, non si dovrebbero incontrare problemi hardware o incomprensioni; il CPC464 sta semplicemente attendendo di essere programmato.

Se si commette un errore nell'inserimento di questo programma, si veda il paragrafo 1.2.7 per vedere se è possibile correggere il programma invece di reinserirlo.

Inserendo il programma della prossima pagina non importa se si utilizzano lettere minuscole (a b c) o maiuscole (**SHIFT**ate - A B C). il computer accetterà comunque il programma. E' NECESSARIO delimitare le parole con degli spazi o altri delimitatori (virgole, due punti, ecc a seconda del caso) nelle posizioni indicate, dal momento che il BASIC AMSTRAD permette di usare le parole riservate (elencate nell'Appendice VIII) all'interno dei nomi di variabili.

La tastiera "fisica" è riportata in figura 4: è chiamata "fisica" perchè molti tasti possono essere ridefiniti dall'utente mediante una serie di token di espansione descritti nelle prossime parti.

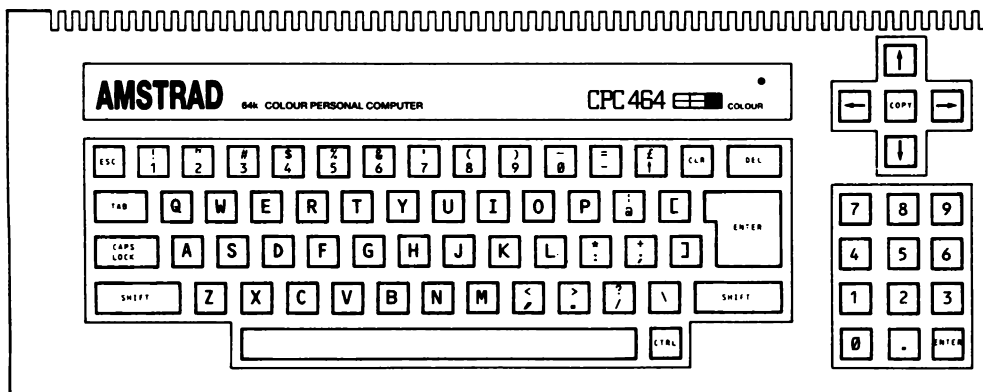


Figura 4: la tastiera del CPC464

Premendo **[ENTER]** la linea di comando o di programma appena immessa verrà accettata dal computer e verranno eseguite le istruzioni in essa contenute - o, se la linea inizia con un numero, essa verrà memorizzata come parte del programma.

[ENTER] corrisponde al tasto "Return" o "Carriage Return" che fa riferimento ai primi tipi di terminali che erano composti da una meccanica simile a quella di una macchina per scrivere. Il termine è così rimasto ed è stato inserito in forma contratta nel set di caratteri ASCII, dove il codice relativo al tasto **[ENTER]** è denotato dalle lettere 'CR'. Si inserisca:

```
10 FOR n=32 TO 255
20 PRINT CHR$(n);
30 NEXT n
RUN
```

Si guardi cosa è accaduto allo schermo:

```
 Amstrad 64K Microcomputer  (v1)

 ©1984 Amstrad Consumer Electronics plc
           and Locomotive Software Ltd.

 BASIC 1.0

Ready
10 FOR N = 32 to 255
20 PRINT CHR$(N);
30 NEXT N
run
 !"#$%&'()*+,-./0123456789:;<=>?@ABCDEFG
HIJKLMNOPQRSTUVWXYZ[\]↑_`abcdefghijklmno
pqrstuvwxyz{|}~
Ready
█
```

Con il breve programma inserito, si è chiesto al computer di visualizzare tutti i caratteri utilizzabili. Se ciò non avviene, è probabile si sia fatto un errore nell'inserimento del programma e che non lo si sia notato; per vedere come risolvere il problema si passi al paragrafo 1.2.7.

Assumendo che tutto sia andato per il verso giusto e che si sia ottenuto il risultato corretto, esaminiamo il programma; ci aiuterà a comprendere come il CPC464 comunica mediante il suo set di caratteri.

La prima cosa da notare è che non si è chiesto al computer di stampare i caratteri con una istruzione come PRINT "abcdefghijklm.....ecc" ma:

PRINT CHR$(n)

n rappresenta semplicemente una variabile; la scelta della lettera è arbitraria e la n è semplicemente la preferita dai matematici in questo campo. Una variabile è un oggetto che contiene una informazione che può variare a seconda delle istruzioni che compongono il programma.

Un numero come 5 è fissato e si trova tra il 4 ed il 6 e quindi non è una variabile; un carattere n è anch'esso fissato (è una lettera dell'alfabeto).

Così come fà il computer a distinguerle? Se la lettera n è intesa come carattere alfabetico, dobbiamo racchiuderla tra virgolette:

"n"

Ed il computer avrebbe risposto con il messaggio Syntax error in quanto non può comprendere la sequenza FOR "n".

Usando la n in tal modo, abbiamo detto al computer che n è una variabile. Per definizione, in BASIC il comando FOR deve essere seguito dal nome di una variabile e quindi il computer assume che qualunque cosa segua un FOR lo sia.

Abbiamo anche detto al computer n=32 to 255. Abbiamo così stabilito che la variabile può assumere valori in sequenza da 32 a 255.

Dopo aver dichiarato questa variabile, dobbiamo dire al computer cosa farne; la linea successiva fa proprio questo:

```
20 PRINT CHR$(n);
```

Questa linea dice al computer di convertire il valore numerico assegnato ad n nel carattere con il numero corrispondente; la funzione CHR$(n) del BASIC effettua proprio questa operazione. Dopo aver cercato in memoria a quale carattere corrisponde il valore di n, il computer lo stampa sullo schermo.

Il punto e virgola alla fine di questa riga, indica al computer di non andare a capo, come accadrebbe normalmente, in modo che i caratteri vengano stampati uno in fianco all'altro e non uno sotto l'altro.

La linea successiva indica al computer che, quando ha effettuato l'operazione con il primo numero della sequenza, deve tornare alla linea dove si trova il FOR ed effettuare la stessa operazione con il prossimo (NEXT) valore assegnato alla variabile n. Questo processo è noto con il nome di ciclo ed è uno degli aspetti più importanti dell'utilizzo e della programmazione dei computer.

Questo ciclo FOR è una delle caratteristiche fondamentali nel campo dei calcolatori ed è presente in varie forme in tutti i linguaggi di programmazione. Esso permette di evitare di immettere lunghe e ripetitive sequenze di comandi e si imparerà presto ad utilizzarlo nei propri programmi.

Quando il ciclo FOR raggiunge il limite specificato (255), l'operazione termina ed il computer prosegue l'esecuzione con la prima linea dopo la linea 30; ma non ve sono e dunque viene terminata l'esecuzione e viene ripresentato il prompt Ready. Il prompt indica che il computer è pronto (ready) ad accettare ulteriori informazioni; è anche possibile immettere nuovamente un RUN e ripetere l'esecuzione del programma. Il programma è memorizzato nel computer e vi rimarrà fino a quando non si dice al computer di memorizzarlo altrove o si spegne il computer; in tal caso i dati (programmi, variabili, ecc) verranno persi a meno che siano stati salvati su cassetta.

Questo programma illustra un aspetto fondamentale dei computer: tutto quello che essi fanno in relazione con i numeri. Il computer ha visualizzato l'alfabeto (insieme ad altri caratteri) usando un numero per identificare ciascun carattere. Quando si preme il tasto A, non si chiede al computer di stampare una A sullo schermo ma di ricercare nella sua memoria le informazioni necessarie a stampare la lettera A sullo schermo. L'effettiva posizione di questo dato è definita dal codice numerico attivato dalla pressione del tasto.

Ogni carattere corrisponde ad un numero; questi sono elencati nell'Appendice III del manuale.

In modo simile i caratteri visualizzati non hanno nulla a che vedere con la scrittura della lettera sullo schermo.

1.2.4

Non ci si preoccupi se non si comprendono tutti i termini tecnici o il gergo usati in questa pagina. E' importante spiegare in modo completo come il computer gestisce le istruzioni e presenti i risultati richiesti, ma è anche probabile che solo coloro che hanno conoscenze tecniche più approfondite possano apprezzare completamente la spiegazione. Se si trova questa parte troppo difficile, si passi al paragrafo 1.2.5.

Ad esempio, il codice della lettera A è 97. Il computer non può comprendere il numero 97 così com' è quindi il numero deve essere convertito in un formato gestibile dal computer: il codice macchina. I principi riguardanti quest'aspetto della macchina sono l'argomento dell'Appendice II.

Al primo impatto, la traduzione dal codice decimale a quello esadecimale potrà risultare piuttosto difficile. Pensare a numeri basati su dieci unità è così naturale che fare in un altro modo è come cercare di mangiare con il coltello e la forchetta scambiati nelle mani.

Occorre un buon grado di apertura mentale per comprendere la notazione esadecimale (abbreviata in ESA), ma una volta compresa questa, molte cose riguardanti i calcolatori e la struttura del sistema di numerazione diverranno più chiare .

Una volta che il computer ha convertito la pressione del tasto A nel tipo di numero da esso conosciuto lo ricerca nella parte di memoria indicata ed il risultato è un'altra serie di numeri che definiscono il carattere. In pratica il carattere visualizzato sullo schermo è costituito da un blocco di dati, memorizzato in forma di matrice numerica.

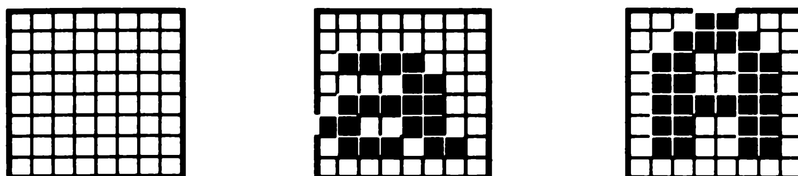


Figura 5: Matrice vuota di un carattere, a minuscola e A maiuscola.

Gli elementi della matrice sono righe e colonne di punti. Il carattere viene visualizzato accendendo o spegnendo i punti appropriati; ogni punto è determinato dai dati memorizzati nella memoria del computer. Vi sono 8 righe di 8 punti in ogni carattere del CPC464 e, se non si trova il carattere desiderato nel set di 256 di cui è dotata la macchina, è possibile definirne di propri usando le parole chiave SYMBOL e SYMBOL AFTER presentate nel Capitolo 8.

I caratteri sono definibili dall'utente usando una qualunque combinazione dei 64 punti che costituiscono la matrice. Perciò, costruendo tutte le possibili combinazioni dei punti di un carattere, è possibile creare molti più elementi (caratteri). Si aggiunga a ciò il fatto che è possibile raggruppare diversi elementi per formare blocchi di maggiori dimensioni e si vedrà che le possibilità dei caratteri definibili dall'utente sono solo limitate dal tempo che si ha a disposizione e dal proprio ingegno.

1.2.5 Torniamo al programma

Il risultato del primo programma inserito era veramente poco fine. Vi sono ancora i resti del messaggio iniziale in cima allo schermo. E' conveniente cancellare lo schermo prima di eseguire il programma. Aggiungiamo al programma una linea che fa proprio questa operazione.

Nella posizione dove si trova il cursore, si inserisca questa linea (il cursore è il rettangolo posto immediatamente a sinistra della parola Ready). Se non si conosce questa nozione conviene ritornare a leggere il corso introduttivo.

```
5 CLS
RUN
```

Si vedrà che lo schermo verrà completamente pulito prima che il set di caratteri venga scritto a partire dall'angolo in alto a sinistra.

Questo dimostra inoltre uno degli aspetti più importanti della programmazione BASIC: non importa il momento in cui si inseriscono le linee di programma, e non occorre aver visualizzato il programma per inserirvi una linea.

Il computer ordina sempre secondo i numeri di linea prima di iniziare l'esecuzione del programma. Si provi a controllare il programma con l'istruzione LIST.

1.2.6 LISTati

E' semplice chiedere al computer il programma che ha memorizzato. Il comando è :

```
LIST
```

ed il risultato sullo schermo :

```
5 CLS
10 FOR n=32 TO 255
20 PRINT CHR$(n);
30 NEXT n
Ready
```

Questo programma rimarrà nella memoria del computer finchè:

* non lo si spegne
* non lo si reinizializza con una sequenza **[CTRL][SHIFT][ESCAPE]** tenendo premuti i tasti finchè avviene la reinizializzazione.
* non si carica (LOAD) o esegue (RUN) un programma da cassetta.
* non si immette NEW **[ENTER]** che cancella il programma e tutte le variabili senza però cambiare la modalità dello schermo o i colori.

Proviamo ora a fare in modo che premendo un tasto funzione vengano eseguite le operazioni **[ENTER]**CLS:LIST**[ENTER]** - una possibilità che velocizza notevolmente l'immissione e lo sviluppo dei programmi. Si immetta:

```
KEY 138, CHR$(13)+"CLS:LIST"+CHR$(13)
```

Si prema ora il punto decimale sotto al tastierino numerico. E' possibile programmare in questo modo fino a 32 tasti ed è possibile ridefinire qualunque tasto della tastiera, nel caso si voglia usare il tastierino numerico per la sua funzione originale; si veda la descrizione del comando KEY nel Capitolo 8

Se il programma è lungo, si può definire il tasto come:

```
KEY 138, CHR$(13)+"CLS:LIST"
```

in modo sia possibile inserire un insieme di linee che si vogliono visualizzare, mentre premendolo due volte si otterrà tutto il listato.

Se si stanno facendo esperimenti con i colori, è possibile trovarsi in una situazione in cui non è possibile leggere sullo schermo in quanto i caratteri e lo sfondo sono dello stesso colore. In tal caso, se si immette:

```
KEY 138, CHR$(13)+"mode 2:ink 1,0:ink0,9"+chr$(13)
```

....si dovrà premere semplicemente il più piccolo dei due tasti **[ENTER]** (quello presente sul tastierino numerico) per ottenere una combinazione di caratteri visibile (non si perderà il programma presente in memoria)

I codici carattere definibili dall'utente vengono cancellati allo spegnimento del computer. Se dunque si sono programmati dei tasti, sarà conveniente scrivere tali programmazioni in un programma e salvare quest'ultimo su nastro per averlo sempre a disposizione.

1.2.7 Correzioni al programma

E' inevitabile commettere errori mentre si inserisce un programma.
Il CPC464 rende la correzione di tali errori il più semplice possibile, evitando nello stesso tempo il problema della sovrascrittura involontaria dei caratteri che non si volevano cambiare.

I tasti che controllano il movimento del cursore (il rettangolo che indica dove verrà inserito il testo che si immette da tastiera) permettono di attirare l'attenzione del computer sul punto dello schermo in cui si vogliono apportare delle modifiche.

Quando si commette un errore nell'inserimento di una linea, ad esempio:

```
10 FOR N=332 TO 255
```

vi sono alcune possibilità:

1. Si può premere **[ENTER]** e ribattere l'intera linea. La linea sbagliata verrà cancellata dalla memoria e sostituita dalla linea che verrà inserita con lo stesso numero di linea.

2. Si può premere il tasto [<-] e spostare il cursore sul carattere errato:

```
10 FOR N=332 TO 255
```

Si noti che il carattere che si trova alla posizione del cursore viene stampato in inverso. In altre parole, il carattere, che è normalmente del colore del cursore, viene visualizzato con il colore dello schermo in modo da renderlo visibile.

Ora si prema il tasto **[CLR]** (abbreviazione di clear, cancella) ed il carattere alla posizione del cursore verrà cancellato - e la linea si sposta di un carattere per riempire il vuoto:

```
10 FOR N=32 TO 255
```

Si prema nuovamente **[ENTER]** e questa linea verrà memorizzata dal computer. Il cursore non deve necessariamente essere alla fine della riga, il computer terrà conto di tutta la linea senza badare alla posizione del cursore.

3. Si può anche portare il cursore sul carattere immediatamente a destra di quello che si vuole cancellare:

```
10 FOR N = 332  TO 255
```

Ora si prema **[DEL]** (abbreviazione di *delete*, cancella); il carattere alla sinistra del cursore viene cancellato, la linea viene spostata a sinistra di un carattere ed il carattere sotto al cursore non viene modificato.

Si prema **[ENTER]** e la linea verrà memorizzata.

```
10 FOR N = 32 TO 255
```

1.2.8 Riflessioni

I metodi precedentemente visti sono comodi se si è trovato l'errore prima di aver raggiunto la fine della linea ed aver premuto **[ENTER]**. La maggior parte degli errori, tuttavia, capitano senza che ci si accorga e fanno sentire la loro presenza solo al momento dell'esecuzione, in quanto il computer risponde con un messaggio di errore (Appendice VIII).

Alcuni errori provocheranno la comparsa sul video della linea errata, con il cursore posto sul primo carattere a sinistra. In questo caso si possono usare i metodi precedentemente esposti.

Se dopo l'errore non viene presentata la linea errata, occorre LISTare il programma, trovare la causa del problema e correggere l'errore.

1.2.9 Correzioni con l'uso del secondo cursore

Innanzitutto si listi il programma usando LIST. Continueremo ad assumere che si stia lavorando su un breve programma di prova che stia all'interno di un solo schermo.

```
5 CLS
10 FOR n = 32 TO 255
20 PRINT CHRS (n);
30 NEXT n
```

L'errore è nella linea 20 - vi è una S al posto dell'identificatore di stringa $ ($ indica al computer di considerare i caratteri seguenti come testo e non come dati numerici). E' possibile reimmettere la linea 20 dall'inizio o usare l'editor di schermo:

Si tenga premuto uno dei tasti **[SHIFT]** e si prema il tasto cursore con la freccia verso l'alto.

E' possibile battere questa combinazione di tasti per ogni linea o attendere la ripetizione automatica del tasto.Nell'istante in cui si tolgono le dita dai tasti il cursore si ferma e per impadronirsi di questa tecnica sarà sufficiente un po' di pratica. Se si supera la linea, è possibile tornare su di essa con la combinazione di tasti **[SHIFT]**-Freccia in basso.

Facendo queste operazioni, il *"Secondo cursore"* si separa dal cursore principale (al quale assomiglia) e viene portato sulla linea che si intende modificare. Iniziamo con il secondo cursore posto sul primo carattere della linea.

```
5 CLS
10 FOR n = 32 TO 255
20 PRINT CHRS (n);
30 NEXT n
Ready
```



Se si cerca di portare il cursore principale sulla linea da correggere, il computer non riconoscerà tale azione.

Se, facendo in questo modo, si sovrascrive la linea, è possibile rimediare premendo il tasto **[ESC]** *PRIMA* di premere il tasto **[ENTER]** o un tasto funzione che contenga chr$(13). Se si dovesse immettere il comando NEW seguito dal tasto **[ENTER]**, si perderà per sempre il programma - dunque attenzione!

Quando si inizia ad immettere una linea, se si cerca di muoversi oltre i limiti di essa senza premere **[ENTER]**, il computer emetterà un bip per indicare che si stanno oltrepassando i limiti. Dopo aver eliminato il problema premendo **[ENTER]** non si sarà perso nulla del programma, a meno che non si sia immesso un numero di linea valido come primo carattere, nel qual caso la linea corrispondente verrà sostituita dalla linea appena immessa e dovrà quindi essere reimmessa.

Quando il secondo cursore sarà nella posizione corretta, si continui a premere il tasto **[COPY]** finchè il secondo cursore raggiungerà il carattere che si intende modificare. Quando la velocità del secondo cursore sembrerà troppo bassa, si potrà tenere premuto il tasto **[COPY]** ed il cursore si sposterà più rapidamente.

```
5 CLS
10 FOR n = 32 TO 255
20 PRINT CHRS (n);
30 NEXT n
Ready
20 PRINT CHR█
```

A questo punto si può lasciare il tasto **[COPY]** ed immettere il carattere $ - che apparirà sotto al cursore principale che si sposterà a destra di un carattere.

```
20 PRINT CHR$█
```

Ora si deve portare il secondo cursore oltre la S di troppo e per fare ciò occorre tener premuto il tasto **[SHIFT]** e premere una volta il tasto **[->]**. Il secondo cursore si porterà sulla parentesi aperta "(". Si lasci il tasto **[SHIFT]** e si tenga premuto il tasto **[COPY]** fino ad oltrepassare con il secondo cursore la fine della riga. Quindi si prema **[ENTER]** e la linea corretta sostituirà quella errata.

E' possibile combinare queste tecniche copiando l'intera linea ed usando le tecniche di correzione del cursore principale ed i tasti **[CLR]** e **[DEL]** senza **[SHIFT]**. Premendo il tasto **[CTRL]** insieme ad uno dei tasti **[<-]** e **[->]** il cursore viene portato immediatamente all'inizio o alla fine della riga.

Un po' di pratica su queste operazioni dovrebbe aiutare.

Infine è possibile richiamare la linea inserendo:

```
EDIT 20
```

Il computer risponderà con:

```
20 PRINT CHRS (n);
```

A questo punto si possono usare i tasti relativi al cursore principale, **[CLR[** e **[DEL]** come abbiamo già visto e, dopo aver corretto l'errore, premere **[ENTER]**. Se si dovessero incontrare problemi, si prema **[ESC]** e LIST. La linea a cui era stato premuto **[ESC]** non è stata modificata.

Ora si immetta LIST ed apparirà sullo schermo la versione corretta del programma. Se non dovesse essere corretta, occorrerà ritentare!

Fino ad ora abbiamo iniziato ad esplorare le possibilità del CPC464. Vediamo ora altri due modi di funzionamento dello schermo; si immetta:

```
MODE 0
RUN
```

Si noterà che lo schermo prima viene ripulito, poi il programma visualizzerà i caratteri a 20 per riga:

Per tornare allo schermo originale, si immetta:

```
MODE 1
RUN
```

E ci si troverà nella modalità iniziale. Per visualizzare lo schermo ad 80 colonne, si immetta:

```
MODE 2
RUN
```

Siamo solo all'inizio e fino ad ora si sarà forse soddisfatta qualche curiosità iniziale. Gli utilizzatori più esperti cominceranno già a pensare alle conversioni dei propri programmi in modo che essi possano essere eseguiti su questo particolare dialetto di BASIC. Gli utilizzatori che si trovano agli inizi dovranno proseguire con i capitoli introduttivi per una panoramica delle caratteristiche specifiche di questo BASIC.

2 Registratore a cassette

Caricamento e operazioni con il registratore a cassette

Gli argomenti affrontati in questo capitolo sono:

* Similitudini e differenze tra le cassette audio e quelle per dati
* Caricamento ed esecuzione di programmi da cassetta) (nastro Welcome)
* Velocità di trasmissione dati su cassetta
* Come salvare i programmi su cassetta
* Errori di lettura

La memoria del CPC464 è in grado di mantenere un programma memorizzato solo fino a quando il computer è connesso alla corrente e l'unità stessa è accesa: nella terminologia informatica tale memoria è detta 'volatile'. Se si vogliono memorizzare i programmi o i dati quando la corrente è spenta occorre memorizzarli su cassetta (o su un altro mezzo di memorizzazione che non sia la memoria volatile, come ad esempio un disco, opzionale).

2.1 Controlli delle cassette

Sulla destra della tastiera vi è il registratore, figura 2 1. La meccanica di tale registratore è essenzialmente equivalente ad un sistema audio ad eccezione del fatto che il controllo del segnale elettronico è ottimizzato per la di memorizzazione di dati di un computer.

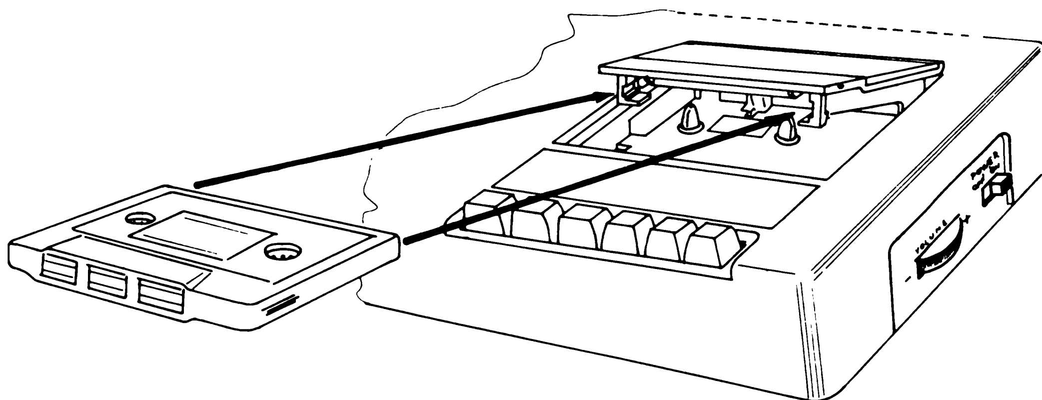


Figura 2.1: Come inserire una cassetta nel registratore.

Analogamente le operazioni della tastierina del registratore sono analoghe a quelle della maggior parte dei registratori audio.

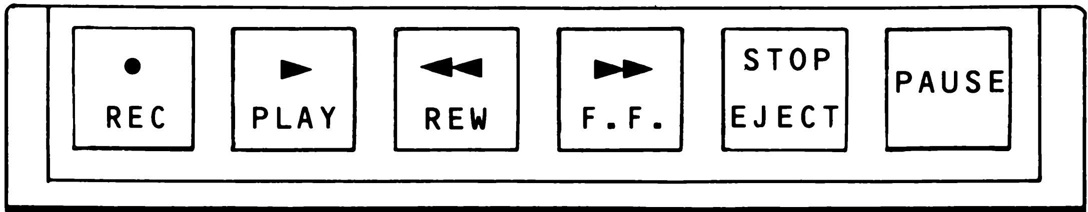


Figura 2.2 I controlli del registratore del CPC464

Si noti che tali tasti vanno premuti più forte rispetto ai tasti di una macchina per scrivere.

[REC] (opera simultaneamente al tasto **[PLAY]** per registrare i dati). Non è possibile attivarlo a meno che non venga inserita una cassetta con la linguetta di 'protezione da scrittura' (si veda la figura della pagina seguente) e non venga chiuso lo sportellino. Per attivare la funzione si deve mantenere premuto il tasto **[REC]** e premere il tasto **[PLAY]**. Il computer scriverà i dati su cassetta quando gli verrà indicato dal programma attualmente in memoria o mediante un comando SAVE inserito da tastiera.

[PLAY] Aziona il meccanismo di trascinamento del nastro per caricare (LOAD) o eseguire (RUN) un programma. Il computer leggerà i dati dalla cassetta, quando verrà indicato dal programma residente in memoria oppure mediante un comando inserito da tastiera. Sia il tasto **[REC]** che **[PLAY]** verranno rilasciati quando il nastro raggiunge la fine.

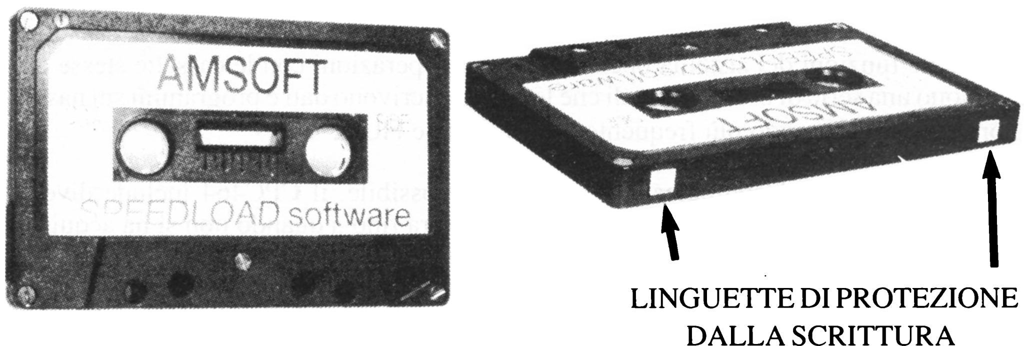
[REW] Riavvolge il nastro dalla bobina destra verso quella sinistra. Non esiste un meccanismo di cancellazione di questa funzione al termine del riavvolgimento del nastro; non si deve quindi lasciare girare il registratore altrimenti la trasmissione del motorino del nastro potrebbe surriscaldarsi quando il nastro si ferma.

[F.F] Fast Forward ovvero avanzamento veloce: farà avanzare il nastro velocemente dalla bobina sinistra verso quella destra. Non esiste un meccanismo di cancellazione di questa funzione al termine del riavvolgimento del nastro; non si deve quindi lasciare girare il registratore altrimenti la trasmissione del motorino del nastro potrebbe surriscaldarsi quandoil nastro si ferma.

[STOP/EJECT] Ferma qualsiasi operazione sulla cassetta e riporta tutti i tasti della tastierina del registratore nella posizione normale. Se questo tasto viene rilasciato e successivamente premuto ancora lo sportellino del registratore si aprirà permettendo di estrarre la cassetta o di inserirne una. La cassetta non può essere estratta fino a quando l'operazione del drive non è cessata.

[PAUSE] Il meccanismo di pausa opera insieme al tasto **PLAY** o ai tasti **[REC][PLAY].** Non deve essere usato durante una lettura o una registrazione di dati in caso contrario verrà indicato un messaggio di errore. Tutte le operazioni di pausa durante la lettura e la registrazione sono gestite dalle istruzioni interne del software del CPC464, e quindi tale comando sarà poco usato.

2.2 Protezione della srittura (figura 2.3)



Tutte le cassette sono dotate di una linghetta di protezione che serve a prevenire cancellazioni involontarie dei dati presenti su esse. Questa linguetta può essere rimossa facendola saltare via mediante un piccolo paio di pinze o un oggetto analogo; in tal modo non si sarà in grado di tener premuto il tasto **[REC]** poichè la cassetta è così protetta dalla scrittura.

Le possibilità di protezione dalla scrittura sono applicabili ad OGNI LATO della cassetta; quindi per proteggere entrambi i lati della cassetta da scrittura occorre rimuovere entrambe le linguette. Nel caso si voglia successivamente ripristinare la possibilità di scrittura sulla cassetta occorre applicare, sul foro che si è creato dopo l'eliminazione della linguetta, del nastro adesivo.

2.3 Caricamento di cassette

La Figura 2.1 illustra come inserire in modo corretto la cassetta fornita insieme al CPC464 all'interno del registratore.

Il nastro della cassetta deve essere completamente riavvolto (dalla bobina destra a quella sinistra): se così non fosse, si prema il tasto **[REW]** fino a quando il nastro viene riavvolto. Se il nastro è uscito dall'apertura della cassetta, prima di inserire quest'ultima nel registratore, occorrerà riportare il nastro all'interno di essa; è comunque possibile in questo modo aver perso delle informazioni memorizzate.

Si noti che mentre è possibile usare nastri che hanno risentito di varie forme di cattivo uso, principalmente pieghe, e altri danni alla superficie del nastro per registrazioni audio non è possibile trattare una registrazione di dati di un computer nello stesso modo ed aspettarsi ancora che continui ad operare in modo affidabile.

Se si danneggia, ad esempio, il nastro formando per sbaglio un nodo ma si è ancora in grado di caricare o eseguire il programma, si deve immediatamente salvare di nuovo il programma su un nastro non danneggiato (mentre il programma è situato in memoria del CPC464) e buttare via la cassetta in modo che non venga la tentazione di usarla di nuovo.

2.4 Esecuzione del nastro Welcome

Il nastro fornito insieme al CPC464 contiene diversi programmi dimostrativi delle capacità sonore e grafiche del computer e del suo software interno (il BASIC AMSTRAD e il Sistema Operativo Macchina (MOS).)

Parte delle funzioni del sistema operativo sono le operazioni con le cassette stesse che includono una serie di brevi comandi che leggono e scrivono dati e programmi sul nastro. I comandi visti in modo più frequente sono LOAD e RUN.

Per rendere l'uso del computer il più semplice possibile, il CPC464 include diverse funzioni speciali che semplificano l'uso della tastiera fino a quando non si ha acquisito dimestichezza con la tastiera. Se si ha acceso il computer si potrà vedere sullo schermo un messaggio iniziale e la parola:

Ready

Si inserisca il nastro come indicato precedentemente (figura 2.1), si azzeri il contatore del nastro **(000)** premendo il piccolo pulsante in fianco alla finestra del nastro; si mantenga premuto il tasto **[CTRL]** e si prema **[ENTER]** situato nell'angolo in alto a destra del tastierino numerico. Verrà visualizzato il messaggio:

RUN"
Press PLAY then any key:

Questo un esempio di programmazione dei tasti, già brevemente introdotti con il comando KEY nel Capitolo 1, dove viene indicato al computer di eseguire una sequenza di comando completa usando una forma di abbreviazione da tastiera la quale serve a velocizzare in modo semplice sequenze di istruzioni richieste con più frequenza.

Si potrebbe scrivere l'istruzione RUN" e premere successivamente il tasto **[ENTER]**, ma è più semplice usare due parole chiave per ottenere lo stesso risultato.

I due tasti **[ENTER]** hanno per la maggior parte delle funzioni lo stesso effetto; in alcune circostanze, comunque, il più piccolo dei due può essere ridefinito in modo da eseguire altre operazioni usando istruzioni del programma. Questo argomento verrà ampliato successivamente nel manuale.

Il tasto **[PLAY]** che viene richiesto di premere è situato davanti al registratore. Lo si prema in modo che resti bloccato. La parte del messaggio 'press any key' (si prema un tasto qualsiasi) è comunemente usata ma può sviare un incauto. E' infatti usata per semplificare le sequenze che indicano al programma di proseguire con l'operazione successiva evitando quindi di specificare ulteriori parole chiave.

Sarebbe stato più corretto dire *'press any key - other than the* **[SHIFT][CAPS LOCK][CTRL][ESC]** *keys (or any keys on the cassette sub-panel)'* (ovvero: si prema un tasto qualsiasi ad eccezione dei tasti **[SHIFT][CAPS LOCK][CTRL][ESC]** o uno dei tasti situati sul pannello del registratore) ma tutti i computer devono assumere alcune abbreviazione nell'interesse della semplicità. Quando in questo manuale (o in altri programmi) viene indicato 'press any key' si assumono le eccezioni prima sottolineate.

Il tasto che verrà premuto non sarà un carattere visualizzato sullo schermoe servirà solo ad azionare il motorino di trascinamento del nastro. Se il nastro non inizia a girare si prema un altro tasto (è buona abitudine premere il tasto **[ENTER]** più grande) e si controlli che non si sia premuto per sbaglio il tasto **[PAUSE]**.

Una volta che il computer ha iniziato a caricare il programma qualsiasi tasto si prema verrà ignorato.

Si noti che non si è specificato alcun nome di programma. Se dopo l'istruzione

RUN"

non viene specificato alcun nome, il computer cercherà il primo programma che trova sul nastro e lo carica. Quando il computer ha trovato il primo programma registrato correttamente su nastro visualizza sullo schermo il seguente messaggio:

Loading WELCOME 1 block 1

Ciò indica che il computer ha trovato la prima serie di blocchi del programma il cui nome è 'Welcome 1'. Ogni programma viene salvato su nastro sotto forma di blocchi di dati (circa 2Kbytes) i quali vengono successivamente letti dal computer: ogni blocco di dati è identificato separatamente sul nastro e il messaggio sullo schermo indica il blocco che il computer sta attualmente leggendo. Dopo ogni blocco di dati il nastro si ferma momentaneamente, poi riprende a girare dopo aver aggiornato il messaggio presente sullo schermo.

Se il computer trova dei dati danneggiati visualizza un messaggio di errore (elencati nell'Appendice VIII) che fornisce la natura dell'errore. L'unica cosa possibile è quella di cercare di eseguire ugualmente il programma fino a quando ciò non viene fatto senza errori.

Assumendo che il nastro sia stato caricato, si leggano le istruzioni sullo schermo, Welcome farà il resto ...

2.5 Supersafe e Speedload

Il CPC464 offre due velocità all'utente: Supersafe a 1000 baud (bit di dati per secondo) e Speedload a 2000 baud. La modalità Speedload scrive e legge quindi ad una velocità doppia rispetto a Supersafe sebbene sacrifica il margine di sicurezza richiesto a causa della scarsa qualità delle cassette a basso costo e degli errori che possono sorgere da un diverso allineamento delle testine sul nastro nei vari registratori.

Per i programmi salvati e caricati sempre dallo stesso registratore, Speedload dovrebbe essere affidabile, sempre che si usino nastri di ottima qualità. Speedload dovrebbe essere inoltre in grado di caricare il software di cassette commerciali senza errori sebbene AMSTRAD avvisa che tale software potrebbe essere registrato usando le opzioni di entrambe le velocità

Il computer si predispone automaticamente alla velocità con cui è stato registrato il nastro. Quando si salva un programma è necessario indicare al computer se si desidera usare Speedload: in caso contrario verrà automaticamente assunto Supersafe.

Per selezionare la velocità più alta per salvare programmi e dati, si controlli che il computer sia in grado di ricevere istruzioni (deve essere visualizzato il prompt Ready) e si scriva:

SPEED WRITE 1

per tornare nella velocità più lenta si può riavviare il computer (in tal caso i dati verranno persi) oppure scrivere (al prompt Ready):

SPEED WRITE 0

2.6 Come salvare programmi e dati su cassetta

BASIC possiede diversi comandi per salvare i dati su cassetta: li abbiamo brevemente riassunti unendo alcuni esempi.

2.6.1 SAVE "<nomefile>"

Il metodo più diretto per salvare dati su cassette consiste nell'usare il comando SAVE quando il CPC464 ha visualizzato il prompt Ready, dopo aver eseguito o listato un programma. Usiamo come esempio il programma visto nel Capitolo 1 che visualizzava i caratteri.

Il <nome file> può essere composto da 16 caratteri (incluso lo spazio). Se si cerca di inserire un nome più lungo il 17esimo carattere e tutti gli altri che seguono verranno ignorati. Con il programma residente in memoria si scriva:

SAVE "CARATTERI"

Il computer visualizzerà il messaggio:

Press REC and PLAY then any key:

Si ricordi ciò che è stato detto riguardo a 'any key'; il nastro inizierà a girare e il computer salverà il programma con il nome CARATTERI.

IMPORTANTE

Il computer non può scoprire se si sono usati o meno i tasti corretti per l'avviamento del registratore, se quindi si è premuto solo il tasto **[PLAY]** il nastro girerà ed il programma visualizzerà un messaggio che indica che il programma è stato salvato mentre invece ciò non è vero.

ATTENZIONE: Se si premono per sbaglio i tasti **[REC]** e **[PLAY]** mentre si voleva invece leggere o caricare il programma il computer cancellerà qualsiasi programma presente sulla cassetta. Se non lo si interrompe premendo il tasto **[ESC]** il nastro continuerà a girare ed i programmi verranno cancellati fino a quando il computer non trova il programma che stava cercando. Se si vogliono evitare questi inconvenienti è buona abitudine eliminare la linguetta di protezione delle cassette.

Vi sono quattro modi per salvare il file del CPC464. Abbiamo appena visto quello generale ma vi sono altre tre alternative per scopi specifici.

2.6.2 SAVE "<nomefile>", A

La procedura è la stessa vista precedentemente ad eccezione del suffisso ,A che indica al computer di salvare il programma di dati in forma di file di testo ASCII.

Questo metodo si applica ai file creati da altri editore altri programmi applicativi; il loro uso verrà discusso successivamente.

2.6.3 SAVE "nomefile",P

Aggiungendo il suffisso ,P si indica al computer di proteggere i dati presenti sulla cassetta in modo che il programma possa essere caricato da chiunque ed essere poi letto, fermandone l'esecuzione con un il tasto **[ESC]**. I programmi salvati in tal modo possono essere richiamati solo mediante un comando RUN o CHAIN.

Se si pensa di aver bisogno successivamente di editare o modificare il programma, sarà necessario effettuare una copia in forma non protetta.

2.6.4 SAVE "nomefile",B, <indirizzo iniziale> , <lunghezza in byte> [,<voce opzionale>]

Questa opzione permette di salvare in forma binaria un blocco di dati presenti in memoria RAM su cassetta. E' necessario indicare al computer dove esattamente inizia la parte di memoria che si vuole salvare, quanto lunga e, se necessario, l'indirizzo della memoria in cui far partire l'esecuzione del file.

Questa caratteristica di salvataggio binario permette di memorizzare uno schermo su cassetta. Uno degli usi principali è quello della costruzione di sequenze di 'titoli' per lunghi programmi: spezzando in tal modo la monotonia di un processo di caricamento.

2.6.5 File a cui non si dato un nome e CAT

Se si salva un file senza fornirgli un nome:

```
SAVE""
```

BASIC lo salverà come Unnamed file (File senza nome). Le cassette possono salvare tanti file con lo stesso nome (inclusi i file senza nome) quanti ve ne stanno sul nastro uno di fianco all'altro, a differenza di un disco che richiede che ad ogni file venga fornito un nome diverso.

L'utente perderà facilmente traccia dei programmi memorizzati se non fornisce loro dei nomi che ricordino il contenuto: e si consiglia di aggiungere anche un codice al nome per ricordare qual'è la versione più recente dei programmi e dei file di dati.

E' possibile CATalogare i contenuti di una cassetta inserendo in comando CAT e seguendo la seguente istruzione:

Press PLAY then any key:

BASIC elencherà il contenuto del nastro visualizzando tutti i nomifile in maiuscolo seguiti dal numero di blocchi ed un singolo carattere che indica il tipo di file :

$ un programma BASIC
% un programma BASIC protetto
* un file di testo ASCII
& un file binario.

Il simbolo Ok al termine della linea indica che il file è leggibile e può quindi essere caricato dal computer. L'esecuzione della funzione CAT non intacca il programma attualmente in memoria.

2.7 Errori di lettura

Se il messaggio Read error (errore di lettura) appare quando il CPC464 sta cercando di caricare un programma o dei dati dalla cassetta, il nastro continuerà a girare ed il computer continuerà a leggere i blocchi che trova dopo l'errore ma non cercherà di caricarli salvo che essi siano stati identificati come blocco 1 del programma che si è cercato di caricare (senza successo).

Ciò significa che dopo un messaggio di errore di lettura è possibile fermare il nastro usando il tasto **[STOP/EJECT]**, e riavvolgere il nastro (con il tasto **[REW]**) all'inizio e premere di nuovo **[PLAY]**. Il computer avrà in tal modo un'altra possibilità di caricare il programma in cui ha trovato l'errore e, con un po' di fortuna, potrebbe aver successo.

Gli errori di lettura possono sorgere per diverse cause, la più comune delle quali, consiste nel danneggiamento del nastro che può essere piegato, o stirato o avere la superficie rovinata. Tali cause possono anche sorgere quando si spegne il computer con la cassetta inserita ed il tasto **[PLAY]** o i tasti **[REC]** e**[PLAY]** abbassati.

Questo perchè quando il tasto è premuto il nastro è ha contatto della testina e quindi è possibile che un breve impulso elettrico possa passare. Anche se il nastro è fermo, questo impulso, che avviene allo spegnimento (o alla accensione) del computer, può effettivamente danneggiare l'informazione contenuta in quella parte di nastro e renderlo non leggibile. Inoltre, il nastro fermo è chiuso ermeticamente tra il traferro della testina ed il rullo di trazione e quindi se il nastro viene mantenuto a lungo in tale posizione si può piegare.

Gli errori di lettura possono anche sorgere durante la momentanea sospensione (**[PAUSE]**) del processo di registrazione o di lettura, o quando è stato originariamente registrato con un altro CPC464 in cui le testine non erano perfettamente allineate.

Gli errori possono anche insorgere a volte per ragioni puramente arbitrarie. Le cassette non sono state originariamente progettate come mezzo di memorizzazione dati ed è per questo che sono imperfette rispetto ai più complessi e più costosi sistemi di memorizzazione 'professionali'.

Ciò nonostante, le cassette hanno eseguito un lavoro eccellente fornendo un 'mezzo' standard a basso costo per memorizzare e reperire informazioni di dati su un computer. La limitazione di carattere fisico della dimensione delle particelle magnetiche sulla superficie del nastro associato alla velocità con cui il nastro passa sotto la testina pone dei limiti alla velocità con cui i dati vengono trasferiti dal nastro al computer. Il tentativo di aumentare la velocità oltre quella tollerata dal sistema porterà ad una operazione poco affidabile: in modo particolare con cassette di software a basso costo.

NOTA: le cassette contenenti programmi di altri tipi di computer non possono essere caricati o letti dal CPC464. Questi potrebbero anche sembrare uguali, potrebbero anche emettere gli stessi 'suoni' se sentiti con un sistema audio, ma non verranno caricati ed eseguiti. Se si trova qualche programma di un altro computer che possa essere caricato ed eseguito saremo lieti di conoscere da voi tutti i dettagli riguardo al computer ed al programma.

2.8 Considerazioni sulle cassette

Sebbene il registratore accetta tutti i tipi di cassetta anche le C90, è necessario usare solo le C12 (sei minuti per lato) o al massimo le C30. I programmi memorizzati alla fine di cassette molto lunghe sono difficili da localizzare a meno che non ci si prepari ad aspettare fino a quando il computer li trovi (e ci si ricordi il nome fornitogli) oppure non si tenga meticolosamente traccia dell'indice del contanastro. Se si vuole riscrivere su un altro programma memorizzato nella cassetta occorre localizzare attentamente il punto in cui inizia il programma da cancellare e stare attenti a non scrivere su programmi che si vogliono tenere.

Soprattutto si usi una cassetta per pochi programmi. Le cassette C12 sono economiche e se si dovessero rovinare si avranno meno tentazioni voler salvare una cassetta buona solo in parte.

Si ricorda infine che il software commerciale viene venduto in strette condizioni di copyright. Non si devono effettuare copie o duplicare software (anche se da dare solo ad amici) fornito su cassette se non entro i termini di condizione di vendita del software (alcuni programmi incoraggiano ad effettuare copie di riserva). Le leggi di copyright sono state riesaminate per contrastare tutte le forme di duplicazione di software non autorizzato, e benchè vi sono stati fino ad ora pochi processi la situazione cambierà sostanzialmente nei prossimi anni e potrebbe essere retroattivamente applicabile.

3 Introduzione al BASIC

Una breve introduzione ai programmi scritti usando il BASIC AMSTRAD

Argomenti trattati in questo libro

* Le regole di sintassi e la descrizione della sintassi
* I comandi di stampa, i canali e la formattazione
* ZONE

3.1 Introduzione al BASIC

La relazione fondamentale che lega il BASIC del CPC464 alle operazioni interne del computer viene introdotta nell'Appendice II. Se non si ha programmato alcun computer, cercheremo di dare un aiuto - anche se sarà necessario fare alcune assunzioni che potrebbero non essere completamente chiare per un principiante. In tal caso suggeriamo di sfogliare un qualunque libro che introduca ai concetti della programmazione.

Si dovrebbe essere in grado di seguire questo capitolo utilizzando anche i semplici esercizi presentati senza necessariamente comprendere tutto quello che succede - anche se imparando più regole diverrà tutto più semplice.

BASIC è il linguaggio che viene fornito insieme al CPC464. E' al suo posto fin da quando si accende il computer e fa sentire la propria presenza con il prompt

```
Ready
```

Il BASIC è il linguaggio più semplice da imparare. E' organizzato mediante parole chiave chiaramente definite ed una grammatica e, opera in modo completamente logico.

Il BASIC AMSTRAD riconosce i comandi elencati nel Capitolo 8. Ogni comando è identificato da una o più parole chiave e può avere un certo numero di parametri - alcuni dei quali sono opzionali. In generale ogni parametro può essere una espressione contenente costanti, variabili e funzioni. Le combinazioni di numeri e lettere sono dette stringhe, e sono utilizzabili varie forme di tipi di dati numerici compresi i tipi decimale, esadecimale e binario.

I file su cassetta vengono gestiti sequenzialmente (uno dopo l'altro) mentre i file ad accesso diretto possono essere scelti in mezzo ad altri senza dover passare per file che non interessano.

3.2 La struttura di un programma BASIC

Le istruzioni di programma vengono presentate al BASIC in forma di linee. Una linea può contenere parecchi comandi, separati da un segno di due punti, e può essere lunga fino a 255 caratteri. Un carattere è un numero, una lettera o uno spazio. In *Modo Diretto* le linee vengono immesse dalla tastiera e non devono iniziare con un numero. In *Modo Programma* le linee vengono ricercate nella memoria e vengono eseguite in sequenza secondo il numero che appare in cima ad esse.

Il BASIC AMSTRAD permette di aggiungere e togliere linee dal programma in Modo diretto e di correggere linee esistenti. Prima di eseguire (RUN) o di listare (LIST) un programma, il BASIC riorganizza internamente le linee secondo la sequenza numerica e senza tener conto del momento in cui ogni linea è stata immessa.

3.3 Input di linea

Il BASIC accetta linee lunghe fino a 255 caratteri, terminate da **[ENTER]**. Durante l'input di una linea, è possibile correggere la linea che si sta immettendo ed usare il secondo cursore per inserire caratteri in qualunque punto dello schermo - vedere il Capitolo 1 Paragrafo 1.2.7.

Tutte le parole chiave devono essere delimitate da un separatore - che può essere uno spazio, un operatore matematico (+, -, ecc.) o un altro carattere riconosciuto come tale. Questo è necessario in quanto è possibile usare variabili il cui nome contenga una parola chiave; ovviamente una parola chiave non può essere considerata come una variabile a meno che essa sia modificata per evitare che il computer la interpreti come una istruzione.

Le parole chiave possono essere immesse in minuscolo o in maiuscolo (mediante il tasto **[SHIFT]**).

Il comando PRINT può essere abbreviato con un punto di domanda ? e, se usato in questa forma non occorre usare alcun delimitatore. Gli operatori matematici (+-*/MOD\) permettono di delimitare una parola chiave e quindi la linea seguente è valida anche se non se ne incoraggia l'uso poichè può condurre a cattive abitudini nell'immissione dei programmi anche quando è necessario inserire degli spazi:

```
for n = 1 to 50:?n:next
```

In modo simile, un apostrofo (**[SHIFT] 7**) può essere sostituito (in un commento) ai caratteri : REM.

Spazi in eccesso verranno ignorati e possono essere usati per "formattare" il testo del programma per indicare aree cicliche, ecc.

3.4 Terminologia

Per descrivere comandi e parole chiave del BASIC deve essere utilizzata una terminologia semplice ma rigorosa. Ogni comando viene presentato come dovrà apparire quando verrà immesso dalla tastiera con ogni variabile o parte opzionale indicata da vari tipi di parentesi che si riferiscono ad oggetti di cui si parlerà nel seguito.

Questi oggetti vengono rappresentati in modi diversi e racchiusi in parentesi angolari. Ad esempio nei posti in cui occorre inserire una espressione che fornisca un valore numerico si troverà:

<espressione numerica>

Tutto quello che non è incluso in parentesi angolari deve essere inserito così com'è. Ad esempio il comando STOP esiste solo nella forma:

STOP

Quando una parte della definizione è opzionale, la parte opzionale viene racchiusa tra parentesi quadre. Ad esempio se una espressione numerica è opzionale, verrà presentata come:

[<espressione numerica>]

Se una parte opzionale può essere ripetuta (e può apparire più volte o non apparire del tutto), dopo la parentesi quadra chiusa apparirà un asterisco. Ad esempio, una stringa di cifre, che deve essere composta da almeno una cifra, apparirà:

<cifra>[<cifra>]*

Esempi di tale espressione possono essere i numeri:

3
34
344
345678
ecc.

In molti posti viene utilizzato un elenco di oggetti separati da virgole. Può essere utilizzata una forma abbreviata illustrata dagli esempi:

elenco di: <espressione> significa: <espressione>[,<espressione>]* o:
elenco di: [#]<numero> significa: [#]<numero>[,[#]<numero>]*

che possono essere oggetti come:

3,4
3,4,4
3,4,5,6,7,8
ecc.

L'elenco può essere composto da un singolo oggetto. Se l'elenco contiene più di un oggetto, ogni oggetto in più deve essere preceduto da una virgola che costituisce il separatore che permette al computer di riconoscere i numeri come oggetti separati.

I numeri possono essere scritti in più modi:

a. <numeri>

b. <numeri con esponente> elevati a potenza, nella forma:

2E4 (2 per 10 elevato alla quarta)

....la parte esponenziale può essere negativa o positiva

c. <numeri con base> numeri binari o esadecimali (vedere Appendice II):

Forma decimale (la condizione normale)	100	
Forma esadecimale	&64 o &H64	(la H è opzionale)
Forma binaria	&X1100100	(la X è necessaria)

3.5 Un po' di pratica - introduzione a PRINT

Per dimostrare come utilizzare la terminologia usata, vediamo alcuni esempi di utilizzo del BASIC.

Una delle parole chiave di BASIC che permettono di utilizzare la maggior parte della terminologia usata è PRINT. Un comando è una parola chiave o una istruzione BASIC che può essere utilizzato sia in modo diretto che in modo programma. Una funzione deve invece essere "chiamata" da un comando:

```
PRINT FRE("")
```

Per fare in modo che il CPC464 possa rispondere ad una domanda, occorre specificare tre cose:

1. Dove si vuole far apparire la risposta - sullo schermo, sulla stampante o da qualche altra parte
2. Occorre dare al computer i dati su cui deve lavorare
3. Occorre dire al computer cosa deve fare con i dati.

PRINT stampa il risultato di un comando su un dato "canale" - dove il canale è un numero da 0 a 9 ed è definito come <espressione di canale>, cioè il numero che definisce il particolare canale che deve essere usato:

0 .. 7 sono canali di testo sulle finestre predisposte con il comando WINDOW.

8 è la porta per la stampante parallela, e può essere utilizzata solo se è stata collegata una stampante compatibile Centronics.

9 invia l'output su un file su cassetta precedentemente aperto dal programma.

Una forma sintetica del comando PRINT (che non usa la forma PRINT USING) è perciò:

```
PRINT [#<espressione di canale>][<elementi da stampare>]
```

Le parentesi quadre stanno ad indicare che non è necessario dichiarare una <espressione di canale>; non è neppure necessario fornire al comando PRINT un elenco di elementi da stampare (in questo caso il computer stamperà una linea vuota). Se non si invierà l'output ad un determinato canale, il CPC464 invierà gli elementi da stampare sul canale #0, il canale relativo allo schermo per difetto. Si provi questa linea ricordando di premere **[ENTER]** per far elaborare la linea dal computer:

```
PRINT "CIAO"
```

Il computer scriverà sullo schermo:

CIAO

Si noti che le virgolette " non vengono stampate sul canale di output. Le virgolette vengono utilizzate dal BASIC esclusivamente per delimitare (indicare l'inizio e la fine di) una stringa costante.

Ora si scriva:

```
PRINT #0 "CIAO"
```

....il risultato sarà uguale

Ma si provi:

```
PRINT #4, "CIAO"
```

....e il computer ha posto il risultato nell'angolo in alto a sinistra dello schermo poichè questa e la prima volta che si manda qualcosa al canale numero 4 (che corrisponde all'intero schermo a meno che sia stato precedentemente definito da un comando WINDOW). La posizione iniziale per il testo su uno schermo è l'angolo in alto a sinistra e il canale numero 4 non era ancora stato utilizzato.

Questa caratteristica del BASIC è particolarmente potente, dal momento che permette di costituire complesse disposizioni di testo sullo schermo usando definizioni e comandi semplici.

Il BASIC stamperà tutto quello che viene racchiuso tra virgolette senza esaminarlo. All'interno di esse è possibile usare anche le parole riservate:

```
PRINT "4*4"
```

produrrà come risultato:

4*4

Per effettuare la moltiplicazione 4 per 4, i numeri e gli operatori (in questo caso il simbolo *) devono essere resi disponibili al BASIC ed inviati ad un canale di output:

```
PRINT 4*4
```

produrrà come risultato

```
 16
```

E' da notare che il numero è spostato di un carattere dal margine sinistro in quanto il BASIC riserva questo spazio per un eventuale *segno meno* (-) che indica un risultato negativo.

Il comando PRINT ha molte altre forme ed ha molte possibilità di formattazione dell'output, utilizzando una serie di modelli.

Gli <elementi da stampare> del comando PRINT sono costituiti da oggetti che devono essere stampati. Essi possono essere costituiti da un'espressione numerica, da una stringa (una variabile stringa predefinita come CIAO$ o qualunque oggetto compreso tra due virgolette) o da una variabile.

PRINT USING dispone i numeri secondo un formato fissato ed in tal modo è possibile allineare numeri ed evitare resti o parti decimali.

3.6 I comandi PRINT USING e ZONE

All'avviamento, il BASIC fissa l'ampiezza di ZONE in 13 colonne. Quando l'istruzione di stampa contiene una virgola (,) l'oggetto che si trova dopo la virgola viene stampato nella prossima posizione di tabulazione, specificato da ZONE. Se sullo schermo ci sono meno colonne di quelle specificate nel comando ZONE, il BASIC stamperà l'oggetto seguente sulla linea successiva. L'oggetto non viene cioè spezzato alla fine della riga corrente.

Se non si specifica alcun formato mediante USING, il BASIC stampa i numeri positivi preceduti da uno spazio ed i numeri negativi preceduti dal meno (-). Tutti i numeri vengono seguiti da uno spazio. Il punto decimale non viene inserito se l'oggetto che deve essere stampato non ha parti frazionali.

Il BASIC AMSTRAD non permette di gestire il tasto **[TAB]** come tasto di incolonnamento, poichè non vi è unità di vedute sull'utilizzo di questo tasto nei vari dialetti di BASIC. Premendo il tasto **[TAB]**, viene stampato il carattere di freccia a destra (lo stesso che si ottiene premendo **[CTRL] I**), ma a parte questo non viene utilizzato nel BASIC AMSTRAD.

3.7 PRINT TAB(<espressione intera>)(<elementi da stampare>)

L'effetto di questo comando è illustrabile più semplicemente attraverso un esempio: si immetta il seguente esempio e si veda il risultato.

```
5 MODE 2: INK 1,0: INK 0,9
10 FOR N = 1 TO 5
20 ZONE 40
30 PRINT TAB (N*4) "HI" ,N
40 NEXT
```

Questo programma illustra sia l'uso di ZONE insieme alla virgola che l'uso della funzione TAB(). Si provi ad eseguirlo sostituendo alla linea 10 la linea:

```
10 FOR N = -5 to 5
```

L'istruzione TAB sposta l'inizio del testo da stampare in avanti del numero di spazi specificato dalla <espressione intera>. ZONE può avere un valore compreso tra 0 e 255 - si vedano le definizioni nel Capitolo 8.

La forma PRINT USING viene usata per formattare il risultato di un calcolo quando si usano numeri reali che potrebbero produrre una serie incontrollabile di cifre decimali. E' un concetto un po' complicato che potrà essere apprezzato solo dopo un certo numero di esempi. La forma completa è:

PRINT[#<espressione di canale>,][<elementi da stampare>][<modello using>][<separatore>]

Questa è certamente una forma difficilmente comprensibile anche perchè il <modello using> si suddivide ulteriormente in:

USING <stringa>;[<lista using>]

e <lista using> si suddivide ulteriormente in:

<espressione>[<separatore><espressione>]*

Si provi il seguente comando:

```
PRINT 123.456, USING "###.##"; 4567.896
```

ed il risultato sarà ...

```
123.456   %4567.90
```

Questo comando illustra molti concetti. Innanzitutto l'oggetto stampato prima della parola USING non è stato sottoposto alla formattazione. Poi il modello using alloca un certo numero di posizioni di "output" che saranno utilizzate per stampare l'oggetto seguente (che può naturalmente essere una variabile). Se il numero supera le dimensioni fissate a sinistra del punto decimale, verrà ancora visualizzato ma preceduto dal simbolo % che indica che si sono dovuti oltrepassare i limiti fissati. Inoltre la virgola posta dopo 123.456 ha portato il numero seguente all'inizio della zona (ZONE) di stampa successiva. Se al suo posto vi fosse stato un punto e virgola, il numero sarebbe stato stampato in fianco ad 123.456, uno spazio alla sua destra. I numeri stampati su una stessa linea sono sempre, per ovvie ragioni, separati da uno spazio.

Infine si noti che l'espressione è stata arrotondata e che le cifre in eccesso non sono state semplicemente eliminate.

Si provi ora:

```
PRINT 123.456, USING "#####.##+";4567.899
```

ed alla fine del numero formattato apparirà il segno +. Un segno negativo precede normalmente un numero negativo.

PRINT USING è una caratteristica molto utile per produrre tabulati. Avvisa inoltre (mediante %) se il formato specificato è troppo restrittivo.

4 Variabili, operatori e dati

Trattamento dell'informazione in un programma BASIC

Gli argomenti affrontati in questo capitolo sono:

* Familiarizzazione
* Tipi di variabile: reale, intera e stringa
* Operatori, espressioni logiche
* Vettori
* DATA

4.1 Scopriamo le parole riservate

Si noti (se non lo si è ancora fatto) che i comandi e le parole riservate nel BASIC AMSTRAD sono delimitate da spazi o segni di punteggiatura, o da operatori numerici. I programmi sono semplici da leggere e correggere in quanto sebbene sia possibile inserire le parole chiave in MAIUSCOLO o in minuscolo, quando il programma viene LISTato tutte le parole chiave vengono convertite in maiuscolo. Se così non fosse vuol dire che si è fatto un errore di battitura ed il programma non verrà eseguito.

Il BASIC AMSTRAD permette di includere nei nomi di variabile le parole chiave. Una variabile nel BASIC AMSTRAD è il nome assegnato dall'utente ad un elemento specifico. Può essere semplicemente una lettera (le variabili devono iniziare sempre con una lettera e mai con un numero) sebbene una variabile il cui nome rifletta l'uso sia più facile da leggere e da capire in un lungo programma:

```
RISPOSTA=4*4: print RISPOSTA
```

Le variabili del BASIC AMSTRAD possono contenere al massimo 40 caratteri (il primo dei quali deve essere una lettera) e sono tutti significativi. Le variabili non possono contenere spazi in caso contrario BASIC leggerà solo le lettere precedenti lo spazio (o lettere e numeri) e quando trova quest'ultimo visualizzerà:

```
Syntax error
```

che indica che è stata trovata una sequenza illegale di caratteri (si veda Appendice VIII). Se, in una frase, si desidera includere due parole separate è possibile usare il punto dove si sarebbe voluto lasciare uno spazio. Tutte le forme usate negli esempi che seguono sono utilizzabili, ricordandosi però di DIMensionare i vettori.

4.2 Abbreviazioni

E' noioso dover scrivere ogni volta PRINT, è possibile invece usare ?: BASIC capirà che si intende PRINT (ciò non vale ovviamente se si pone il punto di domanda tra doppi apici). Si noti che ? non deve essere delimitato da uno spazio come la parola PRINT. Se si scrive la seguente linea di programma:

```
10?4*4
run
```

la risposta sarà la stessa ma se ora si LISTa il programma come per magia appare ...

```
list
10 PRINT 4*4
```

BASIC oltre ad aver convertito il punto di domanda nella parola print ha inoltre inserito lo spazio. In una istruzione PRINT che usa le doppie virgolette...

```
10?"CIAO"
```

E' possibile inoltre omettere su una linea l'ultima doppia virgoletta, come evidenziato dalla abbreviazione **[CTRL][ENTER]** della routine che serve a caricare ed eseguire un programma che visualizza sullo schermo RUN". La stessa cosa è fattibile su una linea di programma sebbene non è una buona abitudine in quanto se si ritorna successivamente sulla linea per aggiungere qualche istruzione ci si dimenticherà, molto probabilmente, di chiudere le virgolette.

4.3 Linee multiple e calcoli misti

E' possibile far eseguire, su una singola linea BASIC, diverse operazioni: una linea infatti può essere lunga al massimo 255 caratteri. Tali istruzioni devono essere separate da due punti:

```
?2*8/5+5-4*777E9/3
```

fornisce come risultato:

```
-1.036E+12
```

E' comunque importante conoscere l'ordine con cui i diversi operatori aritmetici vengono riconosciuti da BASIC al fine di evitare errori fondamentali. Essi sono:

^	Esponenziale - eleva il numero alla potenza di 10
MOD	Modulo - Il resto di una divisione intera
-	Meno unitario (si usa per indicare che il numero è negativo)
*	Moltiplicazione
/	Divisione
\	Divisione intera: restituisce la parte intera di una divisione, la parte decimale viene ignorata
+	Addizione
-	Sottrazione

Vengono eseguite, prima di tutto, le istruzioni racchiuse tra parentesi () e, se il contenuto delle parentesi stesso è in forma mista, viene eseguito seguendo le priorità appena citate; la stessa cosa vale per ulteriori parentesi inserite in quelle presenti. Occorre terminare sempre l'istruzione con tante parentesi chiuse quante erano quelle aperte; in caso contrario verrà visualizzato un messaggio di Syntax error.

4.4 Iniziamo

Abbiamo appreso altre informazioni da quando abbiamo introdotto nel paragrafo 3.5 l'istruzione PRINT. Si dovrebbe aver appreso sufficentemente le regole del BASIC AMSTRAD ed essere in grado di proseguire. Le parole chiave del BASIC verranno introdotte quando necessitano: per la lista completa in ordine alfabetico e la relativa descrizione, nel caso questa non si comprenda dal contesto in cui appaiono, si faccia riferimento al Capitolo 8.

Per molte parole chiave di BASIC (e per chi ha qualche nozione della lingua inglese) il significato è ovvio: il comando GOTO 50, ad esempio significa di proseguire l'esecuzione dalla linea numero 50. END significa 'FINE'; in qualunque punto del programma (anche la prima linea) BASIC trova tale comando, visualizza il prompt Ready.

La modalità diretta (immediata) permette di inserire diverse istruzioni contemporaneamente separandole da il segno due punti :. Una volta comunque eseguita la linea (premendo il tasto **[ENTER]**) l'istruzione viene eseguita e immediatamente dimenticata. E' possibile rieseguirla usando il cursore copia assumendo che questo, dopo l'esecuzione, sia ancora sullo schermo.

4.5 Istruzioni condizionali e logiche

BASIC fa un uso esteso di capacità di effettuare semplici compiti ripetitivi e veloci. Vengono forniti diversi comandi di programmazione che assistono nella generazione di questo processo (ciclo): comandi che iniziano, continuano e scoprono quando finisce un ciclo, se un insieme predifinito di condizioni viene incontrata.

L'ultimo di questi elementi di controllo del ciclo riguarda le 'espressioni relazionali'. In altre parole, la relazione tra due dati viene determinata da un 'operatore relazionale'. E' possibile porre in relazione un dato con un altro oppure mettere in relazione una variabile con un riferimento predefinito. Gli operatori relazionali sono:

<	minore di
<=	minore o uguale
=	uguale
>	maggiore di
>=	maggiore o uguale
<>	diverso

Ecco alcuni programmi dimostrativi su questi operatori che si basano su un argomento a tutti particolarmente caro. Se sullo schermo non si è ancora visualizzato il messaggio:

```
Amstrad 64K Microcomputer (v1)
c 1984 Amstrad Consumer Electronics plc
     and Locomotive Software Ltd

BASIC 1.0

Ready
```

... si premano i tasti **[CTRL][SHIFT][ESC]**, premendoli simultaneamente: il computer verrà riavviato e visualizzerà il precedente messaggio. Si scriva ora il seguente programma (le istruzioni per il loro inserimento sono state date nel paragrafo 1.2.7):

```
10 INPUT "A QUANTO AMMONTA IL TUO STIPENDIO";STIPENDIO
20 IF STIPENDIO < 1000000 THEN GOTO 30 ELSE 40
30 PRINT "RICHIEDI UN AUMENTO DI STIPENDIO":END
40 PRINT " ACQUISTA UNA AUTOMOBILE PIU' GRANDE"
```

Si esegua questo piccolo programma inserendo:

```
RUN
```

e l'**[ENTER]** è obbligatorio. Il computer chiede:

```
A QUANTO AMMONTA IL TUO STIPENDIO?
```

Si noti che il computer aggiunge automaticamente il punto di domanda quando si aspetta dall'utente una risposta. Si risponda alla domanda inserendo solo numeri (e non lettere, simboli di punteggiatura ecc.) e premendo il tasto **[ENTER]**.

Si aggiunga, alla fine di tale operazione e quando riappare il prompt Ready, la linea 5 seguente:

```
5 CLS
```

Eseguendo ancora il programma (RUN) lo schermo viene pulito. Se la propria risposta era inferiore a 1000000 si provi ad inserire un numero superiore per vedere la differenza. Estendiamo ora il programma aggiungendo la linea 50:

```
5 CLS
10 INPUT "A QUANTO AMMONTA IL TUO STIPENDIO";STIPENDIO
20 IF STIPENDIO < 1000000 THEN GOTO 30 ELSE 40
30 PRINT "RICHIEDI UN AUMENTO DI STIPENDIO":END
40 PRINT " ACQUISTA UNA AUTOMOBILE PIU' GRANDE"
50 IF STIPENDIO > 3000000 THEN PRINT "e
che magnifico conto in banca!"
run
```

L'istruzione END nella linea 30 del programma ferma l'esecuzione di quest'ultimo e fa apparire il prompt. Poichè nella linea 40 non vi è nessuna istruzione END il programma procede controllando se la variabile STIPENDIO è maggiore di 3000000; nel qual caso il programma risponde con un migliore augurio!

Proseguiamo ed aggiungiamo un'altra linea:

```
5 CLS
10 INPUT "A QUANTO AMMONTA IL TUO STIPENDIO";STIPENDIO
20 IF STIPENDIO < 1000000 THEN GOTO 30 ELSE 40
30 PRINT "RICHIEDI UN AUMENTO DI STIPENDIO":END
40 PRINT " ACQUISTA UNA AUTOMOBILE PIU' GRANDE"
50 IF STIPENDIO > 3000000 THEN PRINT "e
che magnifico conto in banca!"
60 IF STIPENDIO > 2500000 THEN PRINT
"... prestami diecimila lire"
run
```

Si noti che gli operatori relazionali (> e <) agiscono da delimitatori del numero, non vi è quindi bisogno di inserire uno spazio prima e dopo; nel caso lo si inserisca BASIC lo ignora. Se si fornisce come risposta al programma un numero superiore a 2500000 si potrà notare che BASIC salta la linea 50 ed esegue la linea 60.

Si provi a questo punto, in base a ciò che si è appreso fino a questo momento, a scrivere un programma. Si noti come il programma di esempio sia stato ampliato passo dopo passo; la maggior parte dei programmi si evolvono in tal modo. Si introduce forse, in tal modo, un concetto molto importante della programmazione che utilizza la capacità di organizzazione di BASIC ...

4.6 Evoluzione: l'origine dei programmi.

Il modo con cui BASIC permette di costruire i programmi è una caratteristica molto conveniente. I "puristi" potranno obiettare che tale convenienza porta ad una tecnica di programmazione disordinata e 'poco strutturata' poichè dà la possibilità di aggiungere, così come viene in mente, altre istruzioni; i realisti potrebbero considerare che questo è il modo migliore per ottenere l'interesse dello studente che può vedere tangibilmente i progressi che sta facendo.

Prendiamo ancora il programma precedente di esempio ed aggiungiamo la linea 70 che si crea un ciclo nel programma dopo aver dato il tempo al lettore di leggere il messaggio.

```
5 CLS
10 INPUT "A QUANTO AMMONTA IL TUO STIPENDIO";STIPENDIO
20 IF STIPENDIO < 1000000 THEN GOTO 30 ELSE 40
30 PRINT "RICHIEDI UN AUMENTO DI STIPENDIO":END
40 PRINT " ACQUISTA UNA AUTOMOBILE PIU' GRANDE"
50 IF STIPENDIO > 3000000 THEN PRINT "e
che magnifico conto in banca!"
60 IF STIPENDIO > 2500000 THEN PRINT
"... prestami diecimila lire"
70 for n=1 to 900: next n: goto 5
run
```

Si noti che l'ultima linea e alcune di quelle precedenti sono state inserite in minuscolo per ricordare che BASIC riconosce la variabile e la parola chiave. Si prema due volte il tasto **[ESC]** per fermare l'esecuzione del programma e successivamente si LISTi il programma per vedere come BASIC converte le parole chiave in maiuscolo lasciando la variabile in minuscolo.

La linea 70 introduce un ritardo (mediante un ciclo) poichè il computer conta da 1 a 900 prima di eseguire l'istruzione successiva: GOTO 5. In tal modo il programma continua ad essere eseguito in modo ciclico senza fermarsi mai. L'unico modo per fermarlo consiste nel premere il tasto **[ESC]**: premendolo una volta si ferma il programma. Premendo tale tasto un'altra volta si torna in modalità diretta, senza comunque perdere il contenuto della memoria.

In effetti, salvo che non si prema **[ESC]** mentre il computer sta eseguendo il ciclo della linea 70, il programma termina immediamtamente poichè è in attesa di una risposta. La linea che il computer stava eseguendo in quel momento verrà visualizzata. Se si preme il tasto **[ESC]** mentre il computer attendeva un input verrà visualizzato:

Break in 10

Se lo si preme mentre è nel ciclo di ritardo, dopo la pressione verrà visualizzato:

Break in 70

Se si ferma l'esecuzione all'interno di questo ciclo, si sospenderà l'operazione che può essere ripresa successivamente premendo un qualsiasi tasto. Se invece si fa terminare l'esecuzione del programma questa può essere ripresa con il comando:

CONT

il programma verrà ripreso (CONTinuerà) effettuando l'operazione interrotta.

In questo modo non si perde il programma residente in memoria a meno che ciò non venga esplicitamente richiesto da un comando NEW o riavviando il computer premendo simultaneamente i tasti **[CTRL][SHIFT][ESC]**.

Non c'è quindi bisogno di fornire una uscita di sicurezza per coloro che 'inavvertitamente' riavviano il computer. L'azione di cancellazione della memoria è facoltativo e permanente. Si salvi su cassetta tutto ciò che si pensa possa ancora servire.

4.7 Ulteriori variabili e stringhe

L'essenza del calcolo è la variabile. Si ricordi che se qualsiasi parte di una espressione aritmetica è una variabile anche il risultato sarà variabile.

Le variabili hanno tre caratteristiche o attributi: un nome, un tipo ed una 'organizzazione'. I nomi sono stati spiegati precedentemente (4.1), i tipi sono opzionali quindi possiamo definire una variabile basandoci sulle regole (3.4) come:

<nome>[<tipo>]

I tipi sono:

% per numeri interi dove tutto ciò che segue il punto decimale viene ignorato. Le variabili intere occupano poco spazio in memoria, di conseguenza i programmi che non richiedono il trattamento di parti decimali verranno eseguiti più velocemente se le variabili sono state DEFinite come interi. Il comando DEFINT indica che il nome della variabile senza tipo implicito deve essere considerata intera. Dopo il comando DEFINT A la stessa variabile potrebbe essere chiamata A% o A, entrambe trattano interi. I valori interi sono compresi tra -32768 e +32767.

! indica che la variabile è reale; ciò significa che si deve considerare sia la parte intera che quella decimale. Le variabili reali possono assumere valori compresi tra 2.9E-39 e 1.7E+38.

$ indica che la variabile è una stringa: ciò significa che il contenuto può essere composto sia da lettere che da numeri. In altre parole un insieme di caratteri racchiuso tra doppie virgolette. Ad esempio:

```
NOME$="PAOLO ROSSI"
```

Usando il programma precedente aggiungiamo la linea 6 e modifichiamo la linea 60:

```
5 CLS
6 INPUT "Qual'è il tuo nome; NOME$
10 INPUT "A QUANTO AMMONTA IL TUO STIPENDIO";STIPENDIO
20 IF STIPENDIO < 1000000 THEN GOTO 30 ELSE 40
30 PRINT "RICHIEDI UN AUMENTO DI STIPENDIO":END
40 PRINT " ACQUISTA UNA AUTOMOBILE PIU' GRANDE"
50 IF STIPENDIO > 3000000 THEN PRINT "e
che magnifico conto in banca!"
60 IF STIPENDIO > 2500000 THEN PRINT
"... prestami diecimila lire ";NOME$
70 for n=1 to 900: next n: goto 5
run
```

Si noti che è stato aggiunto dopo lire uno spazio altrimenti il nome dell'utente sarebbe stato scritto attaccato a lire. Se non ci si crede si faccia una prova! Il punto e virgola ; alla fine di una linea PRINT o di quella di INPUT fa in modo che il computer non inizi una nuova linea.

Possiamo anche lavorare con gli interi aggiungendo la linea 61. La si inserisca ed il computer riorganizza il programma:

```
5 CLS
6 INPUT "Qual'è il tuo nome; NOME$
10 INPUT "A QUANTO AMMONTA IL TUO STIPENDIO";STIPENDIO
20 IF STIPENDIO < 1000000 THEN GOTO 30 ELSE 40
30 PRINT "RICHIEDI UN AUMENTO DI STIPENDIO":END
40 PRINT " ACQUISTA UNA AUTOMOBILE PIU' GRANDE"
50 IF STIPENDIO > 3000000 THEN PRINT "e
che magnifico conto in banca!"
60 IF STIPENDIO > 2500000 THEN PRINT
"... prestami diecimila lire ";NOME$
61 CIFRA.GIORNO=STIPENDIO/30:
PRINT "che ammontano a L.";CIFRA.GIORNO;" al giorno"
70 for n=1 to 5000: next n: goto 5
run
```

Si noti che il ritardo della linea 70 è stato incrementato fino a 5000: vi è ora più tempo per leggere ciò che viene visualizzato sullo schermo. Il risultato del calcolo della CIFRA.GIORNO è disordinato; lo si potrebbe arrotondare al valore intero. Si aggiunga la linea 62...

```
5 CLS
6 INPUT "Qual'è il tuo nome; NOME$
10 INPUT "A QUANTO AMMONTA IL TUO STIPENDIO";STIPENDIO
20 IF STIPENDIO < 1000000 THEN GOTO 30 ELSE 40
30 PRINT "RICHIEDI UN AUMENTO DI STIPENDIO":END
40 PRINT " ACQUISTA UNA AUTOOBILE PIU' GRANDE"
50 IF STIPENDIO > 3000000 THEN PRINT "e
che magnifico conto in banca!"
60 IF STIPENDIO > 2500000 THEN PRINT
"... prestami diecimila lire ";NOME$
61 CIFRA.GIORNO=STIPENDIO/30:
PRINT "che ammontano a L.";CIFRA.GIORNO;" al giorno"
62 CIFRA.INTERA%=CIFRA.GIORNO: PRINT
"oppure L.";CIFRA.INTERA%;" ...se non sei
seccato dell'arrotondamento"
70 for n=1 to 5000: next n: goto 5
run
```

Si esegua di nuovo il programma.
Si noti che si deve ricordare di continuare ad usare il tipo % poichè è possibile avere una variabile reale con lo stesso nome di quella intera, la differenza è quindi determinata dal simbolo %. Si può inoltre notare che il computer visualizza delle linee suddivise poichè una sola non è sufficiente a contenere il messaggio: questo accade tre volte. Per scrivere programmi lunghi si usi MODE 2; in tal modo risulta più semplice leggere i programmi.

Per ottenere MODE 2 si scriva semplicemente:

```
MODE 2
```

Per produrre una visualizzazione in bianco e nero che rende la lettura più semplice sul CTM644 si inseriscano i seguenti comandi:

```
INK 1,0
INK 0,13
BORDER 13
```

Si LISTi di nuovo il programma.

4.8 Formato di visualizzazione

Come parte dell'evoluzione del programma è il processo che di volta in volta rende il programma più nitido. La prima cosa che occorre fare consiste nel ri-numerare le linee di 10 in 10 usando il comando RENUM. Al prompt Ready, si scriva:

```
RENUM
```

Si LISTi di nuovo il programma:

```
10 CLS
20 INPUT "Qual'è il tuo nome; NOME$
30 INPUT "A QUANTO AMMONTA IL TUO STIPENDIO";STIPENDIO
40 IF STIPENDIO < 1000000 THEN GOTO 50 ELSE 60
50 PRINT "RICHIEDI UN AUMENTO DI STIPENDIO":END
60 PRINT " ACQUISTA UNA AUTOMOBILE PIU' GRANDE"
70 IF STIPENDIO > 3000000 THEN PRINT
"e che magnifico conto in banca!"
80 IF STIPENDIO > 2500000 THEN PRINT
"... prestami diecimila lire ";NOME$
90 CIFRA.GIORNO=STIPENDIO/30:
PRINT "che ammontano a L.";CIFRA.GIORNO;" al giorno"
100 CIFRA.INTERA%=CIFRA.GIORNO: PRINT
"oppure L.";CIFRA.INTERA%;" ...se non sei
seccato dell'arrotondamento"
110 FOR n=1 TO 5000: NEXT n: GOTO 10
```

Tutti i numeri di linea sono stati modificati compresi quelli all'interno del corpo del programma. Tale opzione non sarebbe conveniente se BASIC non tenesse traccia del numero di linea e non le modificasse contemporaneamente.
Rendiamo ora più ordinata la visualizzazione sullo schermo e, per far ciò eliminiamo momentaneamente il ciclo nella linea 110 inserendo un comando REM:

```
10 CLS
20 INPUT "Qual'è il tuo nome; NOME$
30 INPUT "A QUANTO AMMONTA IL TUO STIPENDIO";STIPENDIO
40 IF STIPENDIO < 1000000 THEN GOTO 50 ELSE 60
50 PRINT "RICHIEDI UN AUMENTO DI STIPENDIO":END
60 PRINT " ACQUISTA UNA AUTOMOBILE PIU' GRANDE"
70 IF STIPENDIO > 3000000 THEN PRINT
"e che magnifico conto in banca!"
80 IF STIPENDIO > 2500000 THEN PRINT
"... prestami diecimila lire ";NOME$
90 CIFRA.GIORNO=STIPENDIO/30:
PRINT "che ammontano a L.";CIFRA.GIORNO;" al giorno"
100 CIFRA.INTERA%=CIFRA.GIORNO: PRINT
"oppure L.";CIFRA.INTERA%;" ...se non sei
seccato dell'arrotondamento"
110 REM FOR n=1 TO 5000: NEXT n: GOTO 10
```

Inserendo il comando REM all'inizio della linea si indica a BASIC di ignorare le istruzioni presenti su quella linea; in tal modo il programma non ha ciclo e si ferma visualizzando il prompt Ready e tutto ciò che ha visualizzato fino a quel momento. Si scriva ora:

```
15 mode 1
```

La linea 15 fisserà il modo di visualizzazione senza tener conto di quello attuale. Ogni volta che si esegue il programma la linea 15 fisserà MODE a 1. Il comando MODE esegue automaticamente un CLS; la linea 10 è quindi ridondante ma la lasciamo comunque.

Eseguiamo ora il programma e rispondiamo:

```
Qual'è il tuo nome? Paolo
A QUANTO AMMONTA IL TUO STIPENDIO? 4000000
PUOI ACQUISTARTI UNA AUTOOBILE PIU' GRANDE
e che magnifico conto in banca!
... prestami diecimila lire Paolo
che ammontano a L. 10958.9041 al giorno
oppure L. 10959
...se non sei seccato dell'arrotondamento
Ready
```

Non è molto elegante; soprattutto per le interruzioni delle parole . Si aggiunga:

```
25 PRINT: PRINT
85 PRINT
```

e si modifichi la linea 100:

```
100 CIFRA.INTERA%=CIFRA.GIORNO: PRINT
"oppure L.";CIFRA.INTERA%;" ...se non sei ": PRINT
"seccato dell'arrotondamento"
```

Si esegua di nuovo il programma; si potrà vedere che il computer ha posto ...se non sei sulla linea precedente poichè ora tale linea viene riempita (in MODE 1). Si aggiunga la linea 120 in modo da far apparire sullo schermo la parola Ready più in basso:

```
120 ?:?:?:?:?
```

Si esegua di nuovo il programma oppure si elimini il prompt Ready scrivendo:

```
120 GOTO 120
```

Una volta eseguito il programma in questa versione è possibile terminarne l'esecuzione mediante il comando **[ESC].** Si ricordi che ? è un modo veloce per scrivere PRINT. Si LISTi il programma per vederne il risultato.

```
10 CLS
15 MODE 1
20 INPUT "Qual'è il tuo nome; NOME$
25 PRINT:PRINT
30 INPUT "A QUANTO AMMONTA IL TUO STIPENDIO";STIPENDIO
40 IF STIPENDIO < 1000000 THEN GOTO 50 ELSE 60
50 PRINT "RICHIEDI UN AUMENTO DI STIPENDIO":END
60 PRINT " ACQUISTA UNA AUTOMOBILE PIU' GRANDE"
70 IF STIPENDIO > 3000000 THEN PRINT
"e che magnifico conto in banca!"
80 IF STIPENDIO > 2500000 THEN PRINT
"... prestami diecimila lire ";NOME$
85 PRINT
90 CIFRA.GIORNO=STIPENDIO/365: PRINT
"che ammontano a L.";CIFRA.GIORNO;" al giorno"
100 CIFRA.INTERA%=CIFRA.GIORNO: PRINT
"oppure L.";CIFRA.INTERA%;" ...se non sei" PRINT:
"seccato dell'arrotondamento"
110 REM FOR n=1 TO 5000: NEXT n: GOTO 10
120 GOTO 120
```

LOCATE

Fino ad ora, la maggior parte dei comandi BASIC sono stati scritti usando la sintassi universale di BASIC che può essere compresa dalla maggior parte dei computer che hanno un interprete. LOCATE è invece una particolare caratteristica del BASIC AMSTRAD (e di diversi altri) che permette di posizionere il cursore (di testo) in una qualunque posizione dello schermo:

```
LOCATE 10,4
```

```
16 LOCATE 10,4
run
```

e si vedrà che il primo input richiesto è abbassato ed inserito di conseguenza. La linea successiva viene portata avanti verso sinistra poichè la linea 25 inserisce due linee vuote. Si prema due volte **[ESC]** e si aggiunga la linea 26:

```
26 LOCATE 10,4
```

Si esegua il programma; la seconda domanda viene scritta sopra la prima. E' ora possibile posizionare il testo nel punto esatto in cui si vuole appaia utilizzando le coordinate elencate nell'Appendice 6.

Se si desidera che tutte le domande e tutti i commenti appaiano sulla stessa linea è necessario far precedere ogni comando LOCATE da un comando CLS: per evitare confusione con l'ultimo oggetto immesso.

Si noti che le coordinate stesse di un comando LOCATE possono essere variabili, generate quindi dal programma. Abbiamo concluso il programma 'Stipendio' baseremo il prossimo esempio su un diverso argomento. Se si vuole salvare il precedente programma si seguano le istruzioni date nel Capitolo 2. Tale programma potrebbe interessare l'utente che vorrebbe magari ampliarlo.

4.10 IF ... THEN

I loro usi sono diretti e piuttosto letterali. IF <espressione logica> THEN GOTO <numero di linea> è una delle tante forme del comando.

IF controlla che il risultato dell'<espressione logica> sia vero nel qual caso esegue l'opzione. L'operazione IF può essere scritta in modo che ripeta un ciclo fino a quando non viene soddisfatta una condizione particolare. Si riavvi il computer usando **[CTRL][SHIFT][ESC]** e si scriva:

```
1 MODE 1
10 AMSTRAD=0
20 PRINT "AMSTRAD CPC464 personal computer a colori"
30 AMSTRAD =AMSTRAD+1
40 IF AMSTRAD <10 GOTO 20
```

Eseguendo tale programma si vedrà che il comando di stampa della linea 20 viene ripetuto fino a quando la condizione della linea 40 è vera. Quindi dopo la linea 40 viene eseguita la 20. Si veda in tal modo il significato del termine 'variabile': il valore di AMSTRAD viene modificato ad ogni ciclo del programma.

Se si vuole vedere cosa accade al valore AMSTRAD nel corso del programma si aggiunga la linea 35:

```
35 LOCATE 1,20: PRINT AMSTRAD: LOCATE 1,AMSTRAD
run
```

Se è troppo veloce si aggiunga un ciclo di ritardo:

```
36 for n=1 to 500: next
```

Aggiungiamo ora un tocco di colore (se si ha l'opzione CTM644) includendo:

```
34 BORDER AMSTRAD
```

Questa linea commuta il colore del bordo nel colore fissato dal valore della variabile AMSTRAD. Si listi ancora il programma:

```
1 MODE 1
10 AMSTRAD=0
20 PRINT "AMSTRAD CPC464 personal computer a colori"
30 AMSTRAD =AMSTRAD+1
34 BORDER AMSTRAD
35 LOCATE 1,20: PRINT AMSTRAD: LOOCATE 1,AMSTRAD
36 for n=1 to 500: next
40 IF AMSTRAD <10 GOTO 20
run
```

E se si vogliono vedere tutti i colori disponibili sul CPC464 si modifichi la linea 40...

```
40 IF AMSTRAD <26 GOTO 20
```

Eseguendo il programma si vedranno tutti i colori disponibili, iniziando con quello più scuro e finendo con quello più chiaro. E' possibile inoltre inserire un messaggio che indichi il numero del colore del bordo attuale aggiungendo nella colonna 35 la parola Bordo:

```
35 LOCATE 1,20: PRINT "Bordo ";AMSTRAD:
LOCATE 1,AMSTRAD
run
```

Quando il programma termina e viene visualizzato il prompt Ready si inserisca:

```
BORDER 14,6
```

Si potrà vedere che il bordo si alterna nei colori 14 e 6. Per saperne di più sui modi grafici ed i colori occorrerà attendere il prossimo capitolo. Per chiudere questo paragrafo si inserisca:

```
ink 1,18,16
```

e ...

```
speed ink 1,5
```

Si cerchi una aspirina e si riavvi il computer. Si ricordi di salvare il programma se si vogliono impressionare gli amici!

4.11 VETTORI

Molti programmi richiedono diverse variabili per memorizzare dati. Ciò va bene ma è poi molto difficile tener traccia di quale variabile è stata usata per memorizzare certi dati. Fortunatamente BASIC ha una soluzione per questa situazione: i vettori di dati.

Cos'è un vettore? Un vettore è principalmente un gruppo di variabili a cui si fa riferimento con lo stesso nome.

Il modo migliore per spiegare ciò consiste nel fare un esempio. Consideriamo un programma che simula un gioco con le carte e che implica la distribuzione delle carte in maniera casuale. Naturalmente la stessa carta non può essere estratta due volte e quindi bisogna tener traccia di ogni carta. Un modo semplice per far ciò potrebbe essere quello di assegnare una variabile ad ogni carta e selezionare quindi la variabile secondo la locazione della carta; 1 ad esempio potrebbe significare che la carta è stata estratta e 0 che è ancora sul tavolo.

Evidentemente ciò comporterà l'uso di 52 variabili separate ed occorre ricordarsi di quale variabile è associata ad una determinata carta: ed è quindi a questo punto che un vettore diventa utile.

Prima di tutto occorre dare un nome al vettore, ad esempio MAZZO. Per accedere ora ad una particolare variabile o elemento del vettore forniamo semplicemente un numero. Quindi i primi tredici elementi sono usati per definire la serie dei cuori, poi i sei sono rappresentati dal MAZZO(6) dieci dal MAZZO(10) ed i re dal MAZZO(13). E' chiaro?

Non è possibile, sfortunatamente continuare a far riferimento ad elementi all'infinito senza indicare con esattezza al computer di riservare spazio in memoria. Il comando DIM viene usato a tale scopo.

Questo comando fissa la DIMensione del vettore ovvero la sua grandezza; nel nostro esempio abbiamo bisogno di un mazzo di carte composto da 52 elementi.

Il comando BASIC è dunque:

```
DIM MAZZO (52)
```

Ciò indica al computer di riservare uno spazio di memoria atto a contenere 52 variabili (effettivamente 53 poichè è possibile accedere anche all'elemento 0).

Possiamo ora scrivere un frammento di una sotto-routine del programma che distribuisce una carta:

```
10 DIM MAZZO(52)
20 FOR X =1 TO 52
30 LET MAZZO(X)=0
40 NEXT X
.......
1000 CARTA=INT(RND*52)+1
1010 IF MAZZO(CARTA) = 1 THEN GOTO 1000
1020 MAZZO(CARTA) = 1
1030 RETURN
```

Si noti che il comando DIM si trova sulla prima linea del programma. Questo perchè un vettore può essere dimensionato solo una volta: non può essere ri-dimensionato successivamente nel corso del programma.

Le linee dalla 20 alla 40 assegnano agli elementi del vettore il valore zero. La sotto-routine che inizia dalla linea 1000 sceglie una carta a caso e controlla che non sia stata già estratta; in tal caso ne sceglie un'altra fino a quando non ne viene selezionata una ancora sul tavolo. La routine modifica poi il valore dell'elemento appropriato del vettore per indicare che quella carta è stata estratta ed esce dalla sotto-routine.

Il trattamento dei vettori non è ristretto alla singola dimensione ma può essere ampliato ad un numero qualsiasi di dimensioni. Ciò si può ottenere aggiungendo alla variabile un ulteriore numero di riferimento. Il vettore (o matrice) 10x10x10, ad esempio, si può fissare con il comando:

```
DIM vettore(10,10,10)
```

Questa tecnica è utile quando si vogliono suddividere i dati in sottoinsiemi di un gruppo più grandi. Nel nostro esempio possiamo suddividere il mazzo in quattro serie di tredici carte alle quali si può accedere separatamente usando il formato:

```
DIM MAZZO (4,13)
```

Se vogliamo cercare ora i quattro di fiori, i quali potrebbero essere stati gli elementi 43 del vettore originale, dobbiamo esaminare semplicemente l'elemento (2,4); assumendo che i fiori si trovino nella seconda 'riga' del nostro nuovo vettore.

I vettori possono essere usati non solo per memorizzare dati numerici ma anche per trattare stringhe. Una applicazione di tale tipo potrebbe essere la registrazione dei nomi delle persone prenotate per un volo o per un posto a teatro.

4.12 DATA

Questo comando, insieme al comando READ, può essere usato per inserire dati nel programma. I dati richiesti sono listati in una linea con ogni elemento separato da una virgola e tutta la linea preceduta dal comando DATA. Ai dati si può accedere in modo sequenziale usando il comando READ.

Un esempio di programma :

```
10 READ X,Y,Z
20 PRINT X;"+";y;"+";Z;"  = ";X+Y+Z
30 DATA 1,3,5
```

I dati possono essere numerici, testo o l'insieme di entrambi. Non ci si preoccupi se i dati non stanno su una linea; è sufficiente iniziare una nuova linea con un comando DATA posto all'inizio. Quando il computer raggiunge un comando READ cerca all'interno del programma il seguente pezzo di dati senza tener conto di dove possa capitare. Ci si assicuri di avere dati sufficienti per tutti i comandi READ in caso contrario verrà visualizzato un messaggio di errore.

L'unico modo per interrompere una sequenza di input di dati consiste nell'usare il comando RESTORE. Questo posiziona il puntatore dei dati all'inizio del programma permettendo in tal modo di rileggere gli stessi dati diverse volte. Il seguente programma illustra l'uso dei comandi DATA, READ e RESTORE:

```
10 FOR C = 1 TO 4
20 READ X$
30 PRINT X$;" ";
40 NEXT C
50 RESTORE
60 GOTO 10
70 DATA CIAO,COME,STAI,OGGI
```

Si prema **[ESC]** per fermare l'esecuzione del programma.

Si noti che sebbene la linea di dati è stata posta alla fine del programma questa poteva essere posta in qualsiasi altro punto.

Il comando DATA non deve necessariamente essere usato solo per contenere informazioni che devono essere visualizzate (PRINT) quando vengono lette; i valori numerici di un comando DATA possono essere, ad esempio, letti in un comando SOUND. Si scriva:

```
10 FOR n=1 TO 30
20 READ s
30 SOUND 1,s,40,5
40 NEXT n
50 DATA 100,90,100,110,120,110,100,0
60 DATA 130,120,110,0,120,110,100,0
70 DATA 100,90,100,110,120,110,100,0
80 DATA 130,0,100,0,120,150
```

Se non si sente alcun suono si regoli il controllo di volume alla destra del computer.

Concludiamo questa breve introduzione al BASIC con un programma che permette di giocare a blackjack (noto anche con i nomi tressette e 21) con il CPC464. Questo programma dimostra molte della caratteristiche di BASIC e dovrebbe essere compreso facilmente grazie all'uso di nomi di variabili rappresentative. E' possibile migliorarlo ulteriormente aggiungendo della grafica e del suono: è generalmente sviluppando il tema che i migliori programmi di BASIC si sviluppano da un modesto nucleo.

Lo scopo del gioco è di chiudere con un totale 21 aggiungendo i valori delle carte che si hanno in mano e per il banco di pareggiare o chiudere senza eccedere il numero 21 e quindi 'andare fuori'. Dopo aver scritto la prima linea si usi il comando AUTO per far numerare automaticamente le linee.

```
1 REM BLACKJACK
10 REM INIZIALIZZAZIONE
20 YC=2:CC=2
30 ASSI=0
40 CACES=0
50 S=0
60 T=0
70 DIM SERIE$(4)
80 SERIE$(1)="FIORI"
90 SERIE&(2)="CUORI"
100 SERIE$(3=="PICCHE"
110 SERIE$(4)="QUADRI"
120 CLS
130 DIM MAZZO (52)
140 FOR X=1TO 52
150 MAZZO (X)=0
160 NEXT X
170 REM DISTRIBUZIONE DI DUE CARTE PER OGNI GIOCATORE
180 LOCATE 10,3
190 PRINT"TU";SPC(15)"BANCO"
200 LOCATE 3,5
210 GOSUB 740
220 S=S+F
230 IF F=11 THE ASSI=ASSI+1
240 LOCATE 3,6
250 GOSUB 740
260 S=S+F
270 IF F=11 THEN ASSI=ASSI+1
280 LOCATE 24,5
290 GOSUB 740
300 T=T+F
310 IF F=11 THEN CACES=CACES+1
320 LOCATE 24,6
330 GOSUB 740
340 T=T+F
350 IF F=11 THEN CACES=CACES+1
360 REM INSERIMENTO OPZIONE CARTA (C) o PASSO (P)
370 X$=INKEY$:IF X$<>"C" AND X$<>"P" THEN 370
380 IF X$="C" THEN 560
390 LOCATE 3,YC+5
400 YC=YC+1
```

```
410 GOSUB 740
420 S=S+F
430 IF F=11 THEN ASSI=ASSI+1
440 REM CONTROLLA PUNTEGGIO E ASSI
450 IF S<22 THEN 370
460 IF ASSI = 0 THEN 500
470 ASSI = ASSI-1
480 S=S-10
490 GOTO 450
500 LOCATE 12,19
510 PRINT "SEI STATO BATTUTO"
520 PRINT:PRINT"UN ALTRO GIOCO (S/N)"
530 X$=INKEY$: IF X$<>"S" AND X$<>"N" THEN 530
540 IF X$="S" THEN RUN
550 END
560 IF T>16 THEN GOTO 700
570 CC=CC+1
580 LOCATE 24,CC+4
590 GOSUB 740
600 T=T+F
610 IF F=11 THE CACES=CACES-1
620 IF T<21 THEN 560
630 IF CACES = 0 THEN 670
640 CACES = CACES-1
650 T=T-10
660 GOTO 620
670 LOCATE 12,19
680 PRINT "HAI VINTO"
690 GOTO 520
700 LOCATE 12,19
710 IF T<S THEN 680
720 PRINT "IL BANCO HA VINTO"
730 GOTO 520
740 REM CARTA
750 LET CARTA=INT (RND(1)*52+1)
760 IF MAZZO(CARTA)=1 THEN GOTO 750
770 MAZZO(CARTA)=1
780 F=CARTA-13*INT(CARTA/13)
790 IF F=0 THEN F=13
800 IF F=1 OR F>10 THEN GOTO 850
810 PRINT F;" di ";
820 IF F>10 THEN F=10
830 PRINT SERIE$(INT((CARTA-1)/13)+1)
840 RETURN
850 IF F=11 THEN PRINT "JACK DI ";
860 IF f=12 THEN PRINT "REGINA DI  ";
870 IF F=13 THEN PRINT "RE DI  ";
880 IF F<>1 THEN GOTO 820
890 F=11
900 PRINT "ASSO DI  ";
910 GOT0 830
```

Si inserisca C per 'vedere la carta' (la carta seguente da aggiungere a quelle originariamente estratte) o P per passare il gioco al banco. Non asseriamo che questa è l'ultima parola nei giochi con le carte per il CPC464, ma pensiamo possa fornire una idea per continuare su questa strada aggiungendo suono e grafica.

(Si noti che è necessario inserire C per carta e P per passo in lettere maiuscole)

4.13 Espressioni logiche

La differenza principale tra una calcolatrice e il computer è la capacità di quest'ultimo di trattare operazioni logiche in applicazioni come le sequenze condizionali IF...THEN. Per far ciò gli operatori logici trattano i valori a cui sono applicati come maschere di bit ed operano sul bit individuale. La descrizione e l'uso è totalmente *logico*: ma è notoriamente difficile descrivere la logica in termini semplici senza la precisione di definizioni concise.

Le due parti di una espressione logica sono note come *argomenti*. Una espressione logica comprende:

<argomento>[<operatore logico><argomento>]

dove:

<argomento> : NOT <argomento>
oppure : <espressione numerica>
o : <espressione relazionale>
o : (<epressione logica>)

Entambi gli argomenti di un operatore logico sono costretti in una rappresentazione intera e se l'intero non è compreso nell'area ammessa viene visualizzato Error 6.
Gli operatori logici, in ordine di precedenza con cui vengono applicati ai bit sono:

AND restituisce 0 a meno che entrambi gli argomenti dei bit siano 1
OR restituisce 1 a meno che entrambi gli argomenti dei bit siano 0
XOR restituisce 1 a meno che entrambi gli argomenti dei bit siano uguali

AND l'operatore logico più usato:

PRINT 10 AND 10

.....darà come risultato 10.

PRINT 10 AND 12

Restituirà 8

PRINT 10 AND 1000

Restituirà ancora 8

Questo a causa del fatto che i numeri 10 e 1000 sono stati convertiti in rappresentazione binaria:

```
      1010
1111101000
```

L'operatore AND controlla ogni corrispondenza di bit alla volta e quando il bit in cima ed in fondo alla riga è 1 la risposta è 1.

```
0000001000
```

```
10 CLS:INPUT "Numero del giorno";giorno
20 INPUT "Numero del mese";mese
30 IF giorno=25 AND mese=12 GOTO 12
40 GOTO 10
50 PRINT "Buon Natale!"
```

Anche OR opera sui bit ma in questo caso il risultato è 1 a meno che entrambi i bit degli argomenti siano 0, nel qual caso il risultato è 0. Usando gli stessi numeri dell'esempio AND:

```
PRINT 1000 OR 10
1002
```

Bit a bit

```
      1010
1111101000
```

Fornisce come risposta:

```
11111010101
```

Ed il seguente programma:

```
10 CLS
20 INPUT "Numero del mese";mese
30 IF mese=12 OR mese=1 OR mese=2 GOTO 50
40 CLS:GOTO 10
50 PRINT "Dev'essere inverno"
```

```
10 CLS
20 INPUT "Numero del mese";mese
30 IF NOT (mese=6 OR mese=7 OR mese=8) GOTO 50
40 CLS:GOTO 10
50 PRINT "Non può essere estate!"
```

L'ultima caratteristica importante consiste nell'unire insieme le diverse condizioni logiche:

```
10 CLS: INPUT "Numero del giorno"; giorno
20 INPUT "Il numero del mese"; mese
30 IF NOT(mese=12 OR mese=1) AND giorno=29 GOTO 50
40 CLS:GOTO 10
50 PRINT "Questo non è nè Dicembre nè Gennaio ma potrebbe essere un anno
bisestile"
```

Il risultato dell'espressione relazionale è 1 o 0. La rappresentazione del bit per 1 è tutti i bit uguali a 1; per 0 tutti i bit sono 0. Il risultato di una operazione logica su due di tali argomenti conterrà 1 per Vero o 0 per Falso.

Controlliamo ciò aggiungendo al programma la linea 60:

```
60 PRINT NOT(mese=12 OR mese=1)
70 PRINT (mese=12 OR mese=1)
```

Quando eseguiamo il programma, se inseriamo 29 come giorno e 5 come mese, si produrrà nella linea 50 la risposta e il valore effettivo verrà restituito alla espressione logica nelle linee 60 e 70.

Infine XOR (*OR esclusivo*) produce un risultato Vero se entrambi gli argomenti sono diversi. La seguente tabella riassume tutte queste caratteristiche.

Argomento A	1	0	1	0
Argomento B	0	1	1	0
risultato AND	0	0	1	0
risultato OR	1	1	1	0
risultato XOR	1	1	0	0

5 Introduzione alla grafica

Analizziamo il colore e la grafica del CPC464

Gli argomenti affrontati in questo capitolo sono:

* Modalità dello schermo e di pixel
* I colori
* INK, PAPER e PEN
* Disegno di linee
* FNESTRE (WINDOW)

5.1 Caratteristiche specifiche della macchina

La descrizione e le applicazioni del BASIC AMSTRAD si sono basate fino ad ora su specifiche standard diverse. Molte operazioni sono puramente aritmetiche e potranno essere eseguite da un 'gergo' di BASIC: comunque, i comandi BASIC che controllano la grafica (e il posizionamento del cursore di testo) sono stati progettati in relazione al modo specifico in cui l'hardware del CPC464 controlla la visualizzazione e dovranno essere compresi bene affinchè si possa ottenere il massimo dal proprio computer.

Come al solito le PAROLE CHIAVE di BASIC vengono visualizzate come lo sono sullo schermo; ulteriori esempi ed una descrizione concisa di queste verranno date nel Capitolo 8.

5.1.1 Scelta del colore

'Nero' cioè il non colore viene considerato come un qualsiasi colore in tutte le seguenti descrizioni che spiegano gli attributi dei colori e i diversi comandi ad essi associati.

Il BORDER può essere fissato con QUALSIASI colore senza tener conto della modalità dello schermo e non viene modificato se viene usato un comando MODE. Può essere lampeggiante o di un colore fisso.

Il numero di colori dell'inchiostro che possono apparire simultaneamente sullo schermo dipendono dal MODE selezionato. Tutti gli inchiostri (INK) possono essere fissati su due colori, cioè lampeggianti, o su un singolo colore. Lo schermo di testo PAPER, la penna (PEN) di testo e quella di grafica possono essere fissati su un qualunque inchiostro disponibile.

5.1.2 Opzioni trasparenti e relazioni tra PEN, INK e PAPER

Ad eccezione delle condizioni in cui vengono specificati due colori per il lampeggiamento, per scrivere sullo schermo si possono selezionare due inchiostri (INK); uno determina il colore della penna (PEN), l'altro quello della carta (PAPER).

N.B. Il numero associato al comando PAPER è l'inchiostro dichiarato per quel numero e **NON** il numero di colore elencato nell'Appendice VI. In modo analogo il numero associato al comando PEN è l'inchiostro dichiarato per quel numero di penna e **NON** il numero di colore elencato nell'Appendice VI.

Il numero per difetto di PAPER è 0, mentre quello di PEN è 1. Per fissare l'inchiostro (INK) per la carta (PAPER) numero 0 a verde, il cui numero di colore è 9 occorre scrivere:

```
INK 0,9
```

In modo analogo, per fissare INK per PEN numero 1 a nero, il cui numero di colore è 0 si dovrà scrivere:

```
INK 1,0
```

Ponendo PAPER e INK allo stesso valore di PEN, cioè INK 0,0 si otterrà tutto lo schermo nero.

Si può rendere il testo 'scritto' trasparente o opaco usando una serie di caratteri di controllo che forniscono un'utile estensione nella maggior parte dei comandi grafici BASIC. Con l'opzione trasparente è possibile ignorare il colore della carta e riscrivere sul grafico, o riscrivere completamente il sottofondo. Il seguente programma illustra questo effetto:

[CTRL][SHIFT][ESC]

```
10 MODE 1
20 INK 2,19
30 DRAW 200,200,2
40 LOCATE 1,21
50 PRINT "1 NORMALE"
60 PRINT CHR$(22)+CHR$(1)
70 ORIGIN 0,0
80 DRAW 500,200,2
90 LOCATE 12,18
100 PRINT "2 TRASPARENTE"
110 PRINT CHR$(22)+CHR$(0)
120 LOCATE 22,15
130 PRINT "3 ANCORA NORMALE"
```

Il primo comando DRAW della linea 30 ha eseguito una modalità trasperente ma il secondo è stato disegnato dopo CHR$(22)+CHR$(1) nella linea 60. Si noti come nel punto di sovrapposizione sia stato modificato il colore e come (in modo trasparente) l'inchiostro riempie il carattere che è stato completamente sovrascritto.

Si scambi l'attivazione della modalità trasparente (linea 60) con la sua disattivazione per vederne l'effetto. Una lista completa di questi comandi addizionali viene fornita nell'Appendice VI.

5.2 Modalità dello schermo

Vi sono tre modalità in cui lo schermo (operazione testo e grafica) può operare:

a) Normale

Mode 1: 40 colonne x 25 linee, 4 INK di testo
320x200 pixel, indirizzabile in 4 colori

b) Multicolore

Mode 0:20 colonne x 25 linee, 2 INK di testo
160x200 pixel, indirizzabile in 16 colori

c) Alta risoluzione

Mode 2: 80 colonne x 25 linee, 2 INK di testo
640x200 pixel, indirizzabile in 2 colori

Si può notare che la differenza consiste nel numero di elementi visualizzabili su una riga.

Ad ognuno dei tre diversi modi di visualizzazione si fa riferimento come schermo o MODE; quest'ultimo può essere visualizzabile, mediante BASIC, solo uno alla volta. Modificando la modalità lo schermo viene pulito (ha lo stesso effetto del comando CLS e di CLG) ma non intacca il contenuto della memoria.

La modifica di tale MODE può essere effettuata dall'interno del programma o in modo diretto.

5.2.1 MODE 0: visualizzazione multicolore

Possono essere visualizzati simultaneamente 16 dei 27 colori disponibili. Lo schermo è composto da 160 pixel per ogni riga e 200 per ogni colonna verticale. Un esempio di tale schermo si trova nell'Appendice VI.

In MODE 0, vi sono 20 caratteri per ognuna delle 25 linee.

5.2.2 MODE 1: standard o modalità per difetto

MODE 1 viene selezionato quando si avvia il CPC464. Possono essere visualizzati contemporaneamente 4 dei 27 colori disponibili sebbene sia possibile passare velocemente a sceglierne altri. Lo schermo è composto da 320 pixel per riga e 200 per colonna. Un esempio si trova nell'Appendice VI.

In MODE 1, vi sono 40 caratteri per ognuna delle 25 linee.

5.2.3 MODE 2: alta risoluzione

MODE 2 permette di visualizzare simultaneamente due colori e usa 80 caratteri di testo per linea: rende il tal modo più semplice da leggere il programma in quanto visualizza pi caratteri.

Lo schermo è composto da 640 pixel orizzontali e 200 pixel verticali.

5.2.4 Si provi ...

Con il computer riavviato **[CTRL][SHIFT][ESC]** si scriva il seguente programma:

```
5 REM ESEMPIO DIMOSTRATIVO DI GRAFICA
10MODE 1
15 INK 2,0
16 INK 3,6: REM SI FISSA IL COLORE USATO NELLA LINEA 90
17 BORDER 1: REM BLU SCURO
20 CLG: REM PULISCE LO SCHERMO
30 b%=RND*5+1 :REM VARIABILI INTERE CASUALI
40 c%=RND*5+1
50 ORIGIN 320,200: REM FISSA L'ORIGINE
60 FOR a = 0 TO 1000 STEP PI/30
70 x%=100*COS(a)
80 MOVE x%,x%: REM SPOSTA IL CURSORE GRAFICO
90 DRAW 200*COS(a/b%),200*SIN(a/c%),3 :REM DISEGNA UNA LINEA
91 IF INKEY$<>"" THEN 20
100 NEXT: REM TORNA ALLA LINEA 60 SE NON VIENE INTERROTTO ALLA
LINEA 91
110 GOTO 20
```

Si esegua il programma (RUN) e si prema un tasto qualsiasi per far disegnare un altro motivo. Ciò mostra alcune caratteristiche importanti dell'hardware e del software del CPC464: questo 'scrive sullo schermo' in modo molto armonico senza sfarfallii o 'strappi' ed il software include effetti molto sofisticati ottenuti con il minimo sforzo. Il comando REM è stato inserito solo per convenienza, serve solo ad includere commenti all'interno del programma.

Si noti che molte linee indicano una immissione successiva - potremmo rendere tali linee più ordinate usando il comando RENUM - questo aiuterà a capire come i programmi vengono sviluppati ed evolvono partendo da una struttura iniziale.

Si salvi il programma su cassetta:

```
SAVE "GRAFICI 5.5.84"
```

Ecco un altro programma dimostrativo che disegna diversi modelli colorati:

```
new
10 a$=INKEY$: REM SI PREMA UN TASTO QUALSIASI PER INIZIARE LA
SEQUENZA DI MODELLI
20 IF a$="" THEN 10
30 CLS
40 m=INT(RND*3): REM SI SELEZIONA UN NUMERO CAUSALE COMPRESO
TRA 0 E 3
50 IF m>2 THEN 40: REM SI PROVA DI NUOVO NEL CASO IL VALORE
ECCEDA 2
60 MODE m
70 i1=RND*26: REM SI SELEZIONA IL VALORE CAUSALE DELL'INCHIO-
STRO
80 i2=RND*26
90 IF ABS(i1-i2)<5 THEN 70
100 INK 0,i1:INK 1,i2
110 s=RND*5+3
120 ORIGIN 320,-100
130 FOR X=-1000 TO 0 STEP s
140 MOVE 0,0
150 DRAW x,300:DRAW 0,600
160 MOVE 0,0
170 DRAW -x,300: DRAW 0,600
180 a$=INKEY$
190 IF a$<>"" THEN 30: REM INTERRUZIONE DEL CICLO PREMENDO UN
QUALSIASI TASTO
200 NEXT x
210 GOTO 10
```

Questo e il precedente programma illustrano un semplice concetto matematico in un modo molto colorato. Entambi effettuano delle somme di numeri generati casualmente per assicurare che ogni modello sia differente.

Il CPC è un ottimo foglio di carta elettronico e uno dei più classici disegni geometrici è la rappresentazione del seno:

```
10 REM DISEGNO DEL GRAFICO DEL SENO
20 MODE 2
30 INK 1,21
40 INK 0,0
50 CLS
60 DEG
70 ORIGIN 0,200
80 FOR n=0 TO 720
90 y=SIN(n)
100 PLOT n*640/720,198*y,1
110 NEXT
```

Il comando PLOT della linea 100 è la parte del programma che disegna una linea. Produce un punto (pixel) sullo schermo ad ogni calcolo del ciclo FOR NEXT (linee 80-110) ed il risultato viene visualizzato sullo schermo.

Il CPC464 ha molti comandi semplici e nel contempo potenti; si può aggiungere un ulteriore effetto visivo aggiungendo al precedente programma la linea:

```
15 BORDER 6,9
```

Si esegua ancora il programma. Il bordo si alterna nei colori 6 e 9. Il passo di lampeggiamento è fissato da valori 'per difetto'. Per far sì che il programma continui ad eseguire il ciclo si prema **[ESC]** per fermare l'esecuzione e si scriva:

```
120 GOTO 50
```

Si noti che il bordo non termina di lampeggiare quando il programma finisce: questo avviene perchè il bordo è controllato in modo indipendente dal resto del programma. Per fermare questo lampeggiamento e fissare il bordo al colore blu, si prema due volte il tasto **[ESC]** e si modifichi come segue la linea 15:

```
15 BORDER 2
```

Si esegua ancora il programma: il lampeggiamento si fermerà.

Per modificare il colore della curva e dello sfondo occorre modificare il colore dell'inchiostro (INK) nelle linee 30 e 40. Quando si LISTa il programma questo dovrebbe apparire come segue:

```
10 REM DISEGNO DEL GRAFICO DEL SENO
15 BORDER 2
20 MODE 2
30 INK 1,2
40 INK 0,20
50 CLS
60 DEG
70 ORIGIN 0,200
80 FOR n=0 TO 720
90 y=SIN(n)
100 PLOT n*640/720,198*y,1
110 NEXT
120 GOTO 50
```

Il numero 1 alla fine della linea 100 indica al computer di disegnare una curva nel colore specificato da INK nella linea di comando 30. Si controlli la definizione di PLOT data nel capitolo 8 e si vedrà come operano esattamente le varie parti di questo comando.

Se si guarda attentamente la curva disegnata sullo schermo si potrà notare che tale linea non è continua, ma suddivisa in molti frammenti piccolissimi. Il più piccolo segmento è un esempio di "pixel" descritto precedentemente.

5.2.5 Cursore grafico e disegno di linee

Abbiamo provato alcuni modi di 'tradurre' programmi in visualizzazioni grafiche; alcuni comandi e concetti sono stati modificati a tal fine. Quando si disegna una linea sullo schermo, vi sono alcune considerazioni importanti da fare per evitare confusioni.

Il primo punto è lo stato attuale della memoria. Il computer ricorda tutti i colori fissati anche dopo un comando NEW che pulisce la memoria. Per riavviare il computer dal punto iniziale occorre usare simultaneamente i tasti **[CTRL][SHIFT][ESC]** (salvando prima, se si desidera, il programma residente in memoria).

Si provi a scrivere:

NEW: CLS

.....dopo aver cancellato il programma precedente. Si scriva ora:

DRAW 100,100

L'istruzione DRAW indica al computer di disegnare una linea partendo dall'ultima locazione del cursore *GRAFICO* fino alle coordinate x,y specificate (100,100). Il CURSORE GRAFICO un concetto labile che indica il punto nel quale verrà posizionato il disegno seguente.

Per sapere dove si trova si possono usare le funzioni XPOS e YPOS. Scrivendo:

PRINT XPOS

verrà visualizzato

100

(che è la posizione x del punto).

Si noti che il testo viene fatto scorrere dall'alto verso il basso; anche la visualizzazione grafica viene spostata ma il cursore grafico resta posizionato nello stesso punto. Si provi a spostare il cursore verso il basso e successivamente si richiedano le posizioni XPOS e YPOS. Il valore è ancora presente in memoria.

Per specificare un colore per il disegno di una linea, si aggiunga tale istruzione al termine del comando DRAW (si faccia riferimento al comando PLOT del programma della precedente pagina).

Occorre prima di tutto specificare l'inchiostro (INK) e si deve ricordare che è possibile specificare solo il numero di inchiostri e colori permessi nella modalità che si sta usando.

Si scriva il seguente programma:

```
10 MODE 1
20 INK 0,10
30 ORIGIN 0,0
40 INK 1,26
50 INK 2,0
60 DRAW 320,400,1
70 DRAW 640,0,2
```

Quello che segue è l'esempio di un programma che usa tutti i concetti spiegati fino ad ora e ne introduce altri molto utili. Si noti come nella prima linea (10) siano state fissate le condizioni relative al colore e all'inchiostro per essere sicuri che qualsiasi parametro presente in memoria non intaccasse l'effetto del programma immesso.

```
10 INK 0,0:INK 1,26: INK 3,18: BORDER 0
20 REM questo programma disegna modelli
30 mode 1:DEG
40 PRINT “Modelli a 3, 4 o 6 lati? “;
50 LINE INPUT p$
60 IF p$=”3" THEN sa=120: GOTO 100
70 IF p$=”4" THEN sa=135: GOTO 100
80 IF p$=”6" THEN sa=150: GOTO 100
90 GOTO 50
100 PRINT : PRINT “Sto calcolando”;
105 IF p$=”3" THEN ORIGIN O,-50,0,640,0,400 ELSE ORIGIN 0,0,0,640,0,400
110 DIM cx(5),cy(5),r(5),lc(5)
120 DIM np(5)
130 px%(5,81),py%(5,81)
140 st=1
150 cx(1)=320:cy(1)=200:r (1)=80
160 FOR st=1 TO 4
170 r(st+1)=r(st)/2
180 NEXT st
190 FOR st=1 TO 5
200         lc(st)=0:np(st)=0
210         np(st)=np(st)+1
220         px%(st,np(st))=r(st)*SIN(lc(st))
230         py%(st,np(st))=r(st)*COS(lc(st))
240         lc(st)=lc(st)+360/r(st)
245         IF (lc(st) MOD 60)=0 THEN PRINT “.”;
250         IF lc(st) < 360 THEN 210
252         px%(st,np(st)+1)=px%(st,1)
254         py%(st,np(st)+1)=py%(st,1)
260 NEXT st
265 CLS: ink 1,2
270 st=1
280 GOSUB 340
290 LOCATE 1,1
300 EVERY 25,1 GOSUB 510
310 EVERY 15,2 GOSUB 550
320 EVERY 5,3 GOSUB 590
33O GOTO 330
340 REM disegna un cerchio pi altri 3,4 o 6 intorno a lui
350 cx%=cx(st):cy%=cy(st):lc(st)=0
360 FOR X%=1 TO NP(ST)
370         MOVE cx%,cy%
```

```
380          DRAW cx%+px%(st,x%),cy%+py%(st,x%),1+(st MOD 3)
390          DRAW cx%+px%(st,x%+1),cy%+py%(st,x%+1),1+(st MOD 3)
400 NEXT x%
410 IF st=5 THEN RETURN
420 lc(st)=0
430 cx(st+1)=cx(st)+1.5*r(st)*SIN(sa+lc(st))
440 cy(st+1)=cy(st)+1.5*r(st)*COS(sa+lc(st))
450 st=st+1
460 GOSUB 340
470 st=st-1
480 lc(st)=lc(st)+2*sa
490 IF (lc(st) MOD 360)<>0 THEN 430
500 RETURN
510 ik(1)=1*RND*25
520 IF ik(1)=ik(2) OR ik(1)=ik(3) THEN 510
530 INK 1,ik(1)
540 RETURN
550 ik(2)=1+RND*25
560 IF ik(2)=ik(1) OR ik(2)=ik(3) THEN 550
570 INK 2,ik(2)
580 RETURN
590 ik(3)=1+RND*25
600 IF ik(3)=ik(1) OR ik(3)=ik(2) THEN 590
610 INK 3,ik(3)
620 RETURN
```

Al momento dell'esecuzione del programma verrà chiesto di inserire un numero, si immetta 3 per risultati più veloci. Il programma visualizzerà successivamente Sto calcolando, e successivamente un punto ogni pochi secondi che indica che il programma è in esecuzione.

La sotto-routine determinata dalle linee 300-320 fa lampeggiare in differenti colori e viene determinata dai comandi EVERY. Se si desidera rallentare il lampeggiamento si modifichino tali linee come segue:

```
300 EVERY 250,1 GOSUB 510
310 EVERY 150,2 GOSUB 550
320 EVERY 50,3 GOSUB 590
```

Per vedere l'effetto ottenuto si consulti nel capitolo 8 la spiegazione del comando EVERY: questa è infatti una delle caratteristiche più utili del BASIC AMSTRAD. Uno degli effetti più interessanti del comando EVERY è il modo in cui accumula richieste nel caso il programma venga fermato con un solo **[ESC].** Si provi a fermare per un momento il programma e poco dopo si prema un qualsiasi tasto. La visualizzazione lampeggia freneticamente in quanto le istruzioni vengono 'accodate'. Vi è solo uno spazio finito di memoria disponibile per tale coda e dunque il comando EVERY viene cancellato quando tale spazio non permette di eseguire altre operazioni.

5.3 Le finestre

L'utente può selezionare otto finestre di testo nelle quali scrivere caratteri ed una finestra di grafica nella quale fare disegni. Le finestre vengono riportate ai valori per difetto quando si fissa una modalità (MODE). Si veda per una ulteriore descrizione il Capitolo 8.

N B: Se la finestra di testo è equivalente allo schermo intero (difetto), la rotazione rapida viene effettuata dall'hardware. Se la finestra di testo è più piccola dello schermo disponibile tale rotazione deve essere effettuata dal software al quale è associata una maggiore lentezza.

Il comando WINDOW definisce i caratteri sinistro/destro/in alto/in basso che definiscono la finestra; le finestre possono essere sovrapposte l'una all'altra e forniscono un veloce modo per riempire rettangoli. Proviamo a scrivere:

```
KEY 139, "mode 2:paper 0:ink 1,0:ink 0,9: list"+chr$(13)
```

Questo fissa il tasto **[ENTER]** più piccolo in modo che cancelli e ripristini il testo in colori visibili nel caso PEN e PAPER vengano posti in combinazioni invisibili. Il seguente programma disegna una serie di finestre sullo schermo ed illustra due punti principali:

```
5 MODE 0
10 FOR n=0 TO 7
20 WINDOW #n,n+1,n+6,n+1,n+6
30 PAPER #n,n+4
40 CLS #n
50 FOR c=1 TO 200: NEXT
60 NEXT
```

Il primo punto è che ogni schermo viene riscritto su quello precedente, ed il secondo fa rilevare che tutti i messaggi appaiono nel canale 0 (a meno che non vengano ridiretti). Si scriva:

```
LIST
```

Il programma verrà visualizzato nel canale 0. Si provi:

```
LIST #5
```

Poi:

```
CLS #6
```

il punto ai cui ci si è indirizzati per l'ultima volta viene scritto su tutti gli altri e il messaggio Ready appare nel canale 0 anche se in quel momento si è mandato il LISTato al canale 5.

Usando il comando WINDOW SWAP nella linea di programma 55:

```
55 IF n=3 THEN WINDOW SWAP 7,0
```
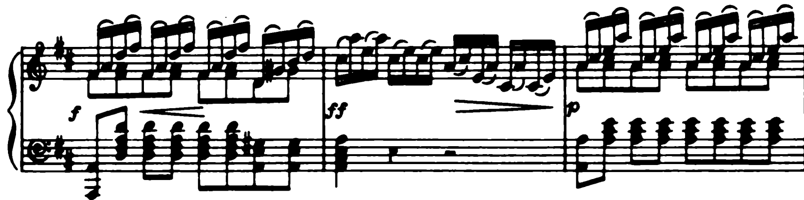
il messaggio Ready verrà inviato, come si sarà immaginato, al canale 7. Si esegua il programma. Sviluppando tale programma si potranno apprezzare i modi di interazione e di utilizzo della finestre.

6 Introduzione al suono

Gli effetti sonori vengono prodotti da un altoparlante contenuto nel computer. Utilizzando il modulatore MP2 ed un apparecchio televisivo, si porti il controllo di volume del TV al minimo.

Il livello del suono può essere regolato con il controllo **VOLUME** sul lato destro del computer. Il suono viene anche inviato ad una presa a cui può essere collegato uno stereo; questa presa si trova sul lato destro del pannello posteriore del computer. In questo modo sarà possibile ascoltare in stereofonia i suoni prodotti dal computer mediante una cuffia o gli altoparlanti dell'impianto hi-fi.

I suoni prodotti dal CPC464 sono di ottimo livello. Per ottenere il meglio dalle possibilità del software di gestione del suono, occorre comprendere la filosofia che si trova sotto la struttura dei tempi.



Argomenti di questo capitolo:

* Periodi dei toni
* Comando Sound
* Inviluppi
* Code e sincronizzazioni

Se si vuole semplicemente che il computer emetta un bip, si scriva

```
PRINT CHR$(7)
```

e non si prosegua ... ma così facendo non si conoscerà una delle caratteristiche più interessanti del CPC464. Questo capitolo costituisce una introduzione e fornisce una panoramica generale, oltre a fornire esempi dettagliati dell'uso dei comandi. Permetterà al programmatore di costruire scale di note, di inventare vari tipi di strumenti musicali e produrre dei brani musicali.

6.1 Aspetti fondamentali del suono

Quando si ascolta un suono prodotto da una nota di un brano di musica, occorre considerare vari aspetti:

1) L'altezza e le variazioni di altezza
2) Il volume e le variazioni di volume
3) La lunghezza della nota.

6.2 Altezza

In una nota musicale, l'altezza è il fattore più importante. Una nota musicale può essere descritta come una oscillazione regolare; ogni oscillazione ha una frequenza, un periodo ed una ampiezza (vedere figura 1).

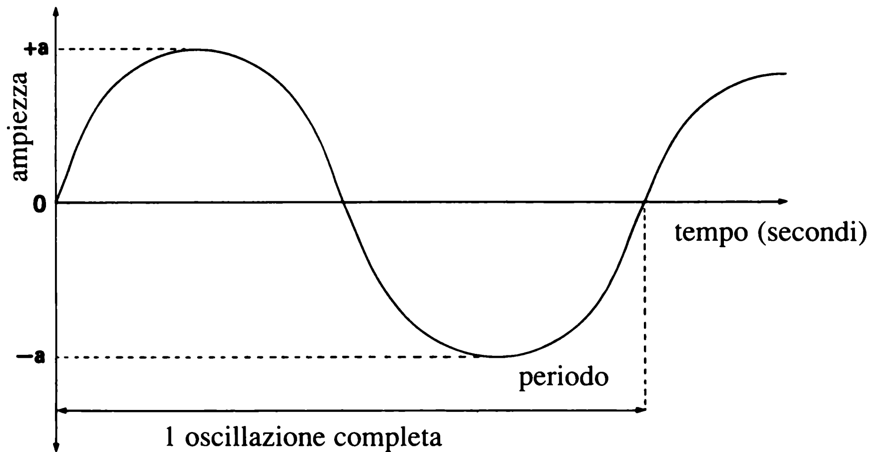


Figura 1: le caratteristiche di una nota

La frequenza è il numero di oscillazioni al secondo ed il periodo è la durata di tali oscillazioni. L'ampiezza è un attributo del volume e non è importante nella definizione dell'altezza di una nota.

La frequenza ed il periodo sono legati da una semplice formula:

Frequenza *(in Hertz)* = 1 / Periodo *(in secondi)*

la relazione tra la frequenza ed il periodo del tono nel comando SOUND è

Frequenza *(in Hertz)* = 125000 / Periodo del tono (in secondi)

Perciò, un periodo del tono uguale a 1000 produce una nota di 125 Hz ed un periodo del tono uguale a 125 produce una nota di 1000 Hz(1 KHz).

Può essere usata una qualunque di queste formule, ma il BASIC AMSTRAD usa la seconda. Non ci si faccia trarre in inganno dal fatto che se il periodo assume valori molto grandi, la frequenza (e quindi l'altezza della nota) si abbassa. L'altezza della nota appare nella descrizione del comando SOUND come <periodo del tono>. La gamma di toni permessa è molto vasta (da 0 a 4095) e deve essere espressa da un intero. Per questo motivo possono sorgere alcuni problemi di arrotondamento nella costruzione delle scale musicali ma nulla che possa offendere anche le orecchie più critiche - vedere Appendice VII.

Quando uno strumento musicale reale produce una nota, l'altezza può variare, a volte deliberatamente con un vibrato. Usando il comando ENT (Tono dell'inviluppo) possiamo progettare un formato specifico per la struttura dei cambiamenti di periodo all'interno di ogni nota - e ciò può essere richiamato con il comando SOUND. Questa caratteristica è nota con il nome di inviluppo del tono.

6.3 Volume

Per fissare il volume iniziale di ogni suono, vi è una semplice scala: aumentando il valore aumenta il volume; il volume è specificato da un intero compreso tra 0 e 15 (tra 0 e 7 se non si specifica l'inviluppo del volume).

Il volume può essere controllato con il comando SOUND.

Se si è già provata qualche semplice nota, si sarà notato che una nota senza alcuna elaborazione non è particolarmente esaltante. Ciò avviene perchè negli strumenti musicali convenzionali, quando una nota viene suonata, all'inizio vi è una crescita del volume della nota (e questa fase è chiamata attacco) seguita da una fase in cui il volume cala (fase di discesa).

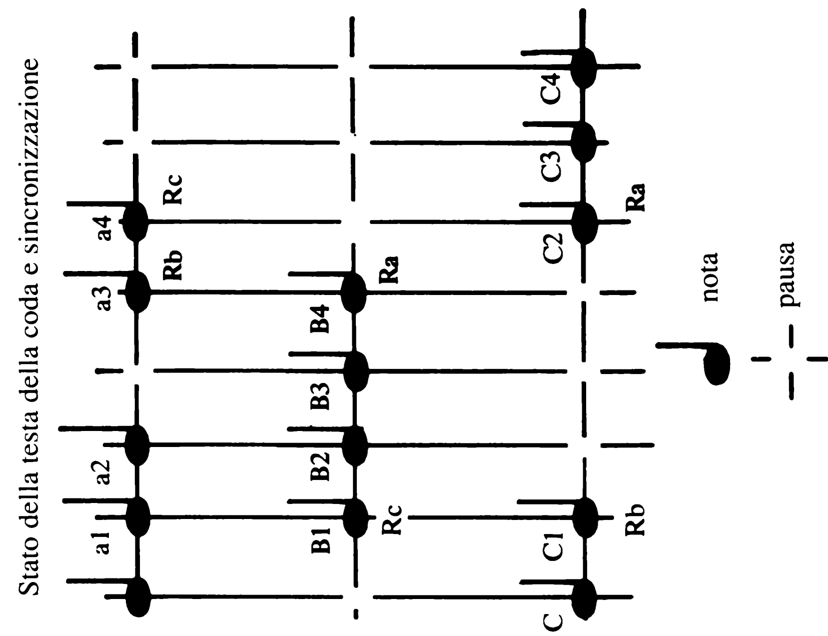
La forma di questo andamento varia da strumento a strumento e può essere simulata su un computer regolando le caratteristiche del volume mediante gli inviluppi del volume.

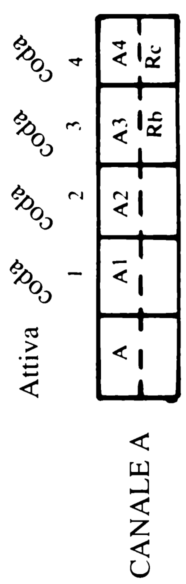
6.4 Lunghezza di una nota

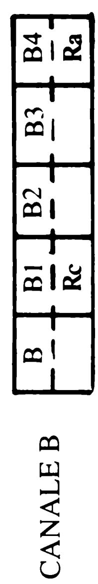
La lunghezza di una nota è la caratteristica principale di qualunque composizione musicale. L'unità di misura di una nota musicale è la semiminima, e vi sono poi le minime, le semicrome, ecc. che sono multipli e frazioni dell'unità base. Tuttavia le musiche possono essere suonate a varie velocità e la durata di una semiminima standard deve ancora essere determinata.

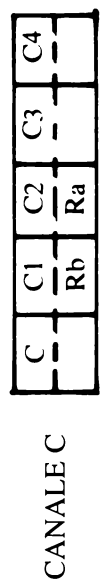
Per questa operazione vi è una espressione all'interno del comando SOUND conosciuta come <durata> che è un intero in cui ogni unità corrisponde ad 1/100 di secondo (cioè 100=1 sec.). Si noti che è possibile determinarla dalla lunghezza dell'inviluppo del volume che evita confusioni se si usa un inviluppo del volume.

Rendezvous dei canali del suono









A–A4
B–B4
C–C4
} comandi SOUND

Ra – Rendezvous con il canale A
Rb – Rendezvous con il canale B
Rc – Rendezvous con il canale C

6.5 Altri suoni

Il rumore casuale ("rumore bianco") è anch'esso un suono e può essere generato dal computer anche se non è una nota musicale. Può essere sommato alle note principali per formare interessanti variazioni o può essere usato per particolari effetti - una esplosione è fondamentalmente del rumore bianco con un controllo sull'invilupo del volume. Questo rumore varia casualmente la frequenza intorno ad un periodo fissato dal parametro <periodo del rumore> del comando SOUND. Un'importante caratteristica consiste nel fatto che, sebbene siano disponibili tre canali separati, con la possibilità di fissare tre <periodi del tono>, può essere fissato un solo <periodo del rumore> che verrà miscelato all'interno della combinazione voluta di canali.

6.6 Suoni multipli e canali

La maggior parte dei brani musicali sono scritti con almeno due voci, una bassa ed una alta. Per permettere questa possibilità nel CPC464 vi sono tre canali chiamati A, B e C. Essi possono suonare indipendentemente o sincronizzati, per coincidere nel punto desiderato. La selezione del canale viene fatta dal parametro <stato del canale> del comando SOUND.

6.7 Code dei canali

Ogni canale ha una coda di suoni da produrre. Nella coda vi è posto per 5 diversi comandi SOUND: uno attivo e quattro in attesa. Il sistema operativo del CPC464 può eseguire altre operazioni, durante l'esecuzione dei comandi della coda ritornando al suono solo per prelevare altri comandi SOUND.

6.8 Stato del canale

La parola chiave SQ permette di conoscere lo stato del canale che si intende interrogare, e riporta informazioni riguardanti le posizioni libere della coda, i rendez-vous e le attese. ON SQ GOSUB è un comando di interruzione (interrupt) usato per riportare l'attenzione del computer alla parte del programma relativa alla produzione del suono.

6.9 Rendezvous ed attese

Per costringere il canale a sincronizzarsi, vi è la possibilità di *Rendezvous*. In questo caso viene posto un segnale su due o più comandi che provoca l'esecuzione simultanea dei comandi SOUND. Questa possibilità è molto utile quando si vuole tornare ad uno stato di sincronizzazione di note lunghe nel punto in cui il brano contiene arresti o attese non simultanee su tutti i canali.

Le attese (vedere <stato del canale>) sono importanti quando occorre effettuare una sincronizzazione di tutti i canali (ad esempio all'inizio di un brano), inserendo un ritardo, ed è necessario per avere tutte le code pronte. Viene perciò usata una sequenza Attendi e Rilascia.

6.10 Sequenza di comando per la generazione del suono

Forma del comando: SOUND G,H,I,J,K,L,M

Dove ..
G: Stato del canale
H: Periodo della nota
I: Durata
J: Volume
K: Inviluppo del volume
L: Inviluppo della nota
M: Periodo del rumore

I parametri del comando SOUND sono tutti numeri interi e solo i primi due sono obbligatori. Gli altri sono opzionali, ma hanno valori per difetto che verranno discussi durante la descrizione del parametro.

6.10.1 Descrizione dei parametri:

G: Stato del canale

Valore: da 1 a 255
se viene omesso assume il valore: nessuno (obbligatorio)

Nel CPC464 è possibile suonare contemporaneamente tre diversi suoni. Vi sono infatti tre diversi canali: A, B e C. Il parametro viene scritto in forma decimale ma viene convertito in un valore intero ad 8 bit (in modo che ogni bit abbia un significato particolare); i bit attivi indicano i seguenti fatti:

DECIMALE	BIT	COMANDO
1	0 LSB	Invia il suono al canale A
2	1	Invia il suono al canale B
4	2	Invia il suono al canale C
8	3	Rendezvous con il canale A
16	4	Rendezvous con il canale B
32	5	Rendezvous con il canale C
64	6	Attendi
128	7 MSB	Svuota

(LSB e MSB significano rispettivamente Least Significant Bit o bit meno significativo e Most Significant Bit o bit più significativo).

esempi:

2 = invia il suono seguente al canale B
5 = invia il suono seguente ai canali A e C
98 = 64+32+2 = invia il suono seguente al canale B, rendezvous con il canale C ed attendi.

E' importante ricordare che se si deve effettuare un rendezvous tra due o più canali, è necessario attivare i rispettivi bit dello <stato del canale> nel momento giusto.

L'attesa di alcuni canali blocca l'esame del comando e blocca la coda di fronte ad esso, fino a che essa verrà liberata da un comando RELEASE (o svuotata da un altro comando SOUND).

Quando viene attivato con uno o più canali il bit di svuotamento, il comando SOUND viene eseguito immediatamente, lasciando la coda vuota ed il canale inattivo. Ogni suono attualmente in esecuzione viene fermato.

H: Periodo della nota

Valore: da 0 a 4095
se viene omesso assume il valore: nessuno (obbligatorio)

Specifica il periodo che determina la frequenza del suono. La frequenza viene ottenuta mediante la formula:

frequenza = 125000-periodo

Con 0 non viene specificata alcuna frequenza; questo uso del comando è utile se occorre utilizzare solo il rumore.

I: Durata

Qualunque intero tra -32768 e 32767
se viene omesso assume il valore: 20

Per valori maggiori di zero, rappresenta la durata della nota in 1/100 di secondo. Quando uguale a zero, la durata viene determinata dall'inviluppo del volume specificato.

Quando la durata è minore di zero, il valore specificato e cambiato di segno specifica il numero di volte che occorre ripetere l'inviluppo del volume.

J: Volume

Qualunque valore intero compreso tra 0 e 15 (o tra 0 e 7 se non si specifica l'inviluppo del volume)
se viene omesso assume il valore: 12 (4 se non si specifica l'inviluppo del volume)

E' il volume iniziale e può essere modificato da un eventuale inviluppo di volume. Zero sta per volume 0.

K: Inviluppo del volume

Qualunque valore intero compreso tra 0 e 15
se viene omesso assume il valore: 0

Indica un inviluppo predefinito. Per definire un inviluppo si usa il comando ENV. Una definizione permanente è costituita dall'inviluppo 0 che non può essere modificata con il comando ENV ed è fissato a 2 secondi costantemente al livello <volume>.

L: Inviluppo della nota

Qualunque valore intero compreso tra 0 e 15
se viene omesso assume il valore: 0

Indica un inviluppo predefinito. Per definire un inviluppo si usa il comando ENT. Una definizione permanente è costituita dall'inviluppo della nota 0 che non può essere modificata con il comando ENT ed è fissato costantemente a <periodo della nota>.

M: Periodo del rumore

Qualunque valore intero compreso tra 0 e 15
se viene omesso assume il valore: 0

Specifica il rumore che deve essere aggiunto al suono. Se viene usato il valore per difetto zero, non viene aggiunto alcun rumore. Occorre ricordarsi che è possibile utilizzare un solo rumore per volta. Ciò significa che ogni canale utilizzato dal rumore riceverà lo stesso rumore.

6.11 Il comando ENV e gli inviluppi del volume

L'attacco e la discesa aggiungono una nuova dimensione alla nota e la rendono più viva. Questo comando controlla l'inviluppo del volume e permette di modellare la forma di una nota. Prima di descrivere la forma al computer, conviene modellare su carta la forma della nota richiesta. Si veda l'esempio qui sotto:

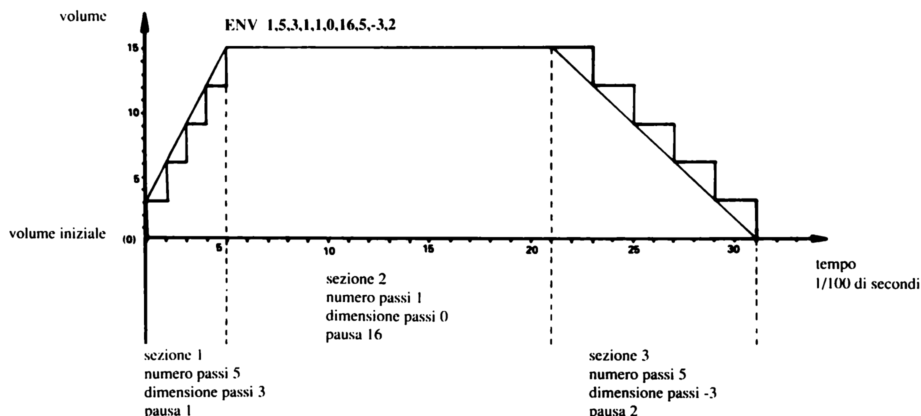


Figura 3: Esempio di inviluppo del volume

La forma della nota deve essere specificata mediante numeri in modo che possa essere immessa in un comando. Per questa operazione, si suddivida la forma in sezioni, al massimo 5. In ogni sezione la forma deve essere una linea diritta. Ora si divida ogni sezione in passi. Il numero di passi scelto è noto come <numero passi>. Questi passi avranno una lunghezza chiamata <pausa> dove 1 = 1/100 di secondo.

I passi avranno anche una crescita o un calo del volume noti come <dimensione passi>. Se non viene specificato alcun passo, non vi è alcuna variazione di volume all'interno della sezione.

NB: Quando si cerca di simulare uno strumento musicale, è conveniente porre il volume iniziale e finale di ogni nota uguali a zero; perciò il valore del volume iniziale nel comando SOUND dovrà essere zero ed il volume verrà fornito solo dalla forma dell'inviluppo.

Forma del comando:

ENV N,P1,Q1,R1,P2,Q2,R2,P3,Q3,R3,P4,Q4,R4,P5,Q5,R5

N: Numero dell'inviluppo	valori tra 1 e 15
P1 .. P5: Numero passi (sezioni 1 .. 5)	valori tra 0 e 127
Q1 .. Q5: Dimensione passi (sezioni 1 .. 5)	valori tra -128 e 127
R1 .. R5: Pausa (sezioni 1 .. 5)	valori tra 0 e 255

Il <numero dell'inviluppo> è obbligatorio. E' necessario specificare completamente una sezione, ma il numero di sezioni è opzionale. Ad esempio, se si omettono le sezioni 4 e 5 verranno assunte solo 3 sezioni.

6.12 Il comando ENT e gli inviluppi dei toni

Hanno esattamente lo stesso tipo di costruzione degli inviluppi del volume, tranne che questi inviluppi provocano invece piccole oscillazioni nella frequenza di una nota; in altre parole una specie di vibrato. Dopo aver deciso la forma della frequenza della nota, la si disegni come si è visto precedentemente e la si divida in sezioni e passi. Si veda l'esempio qui sotto:

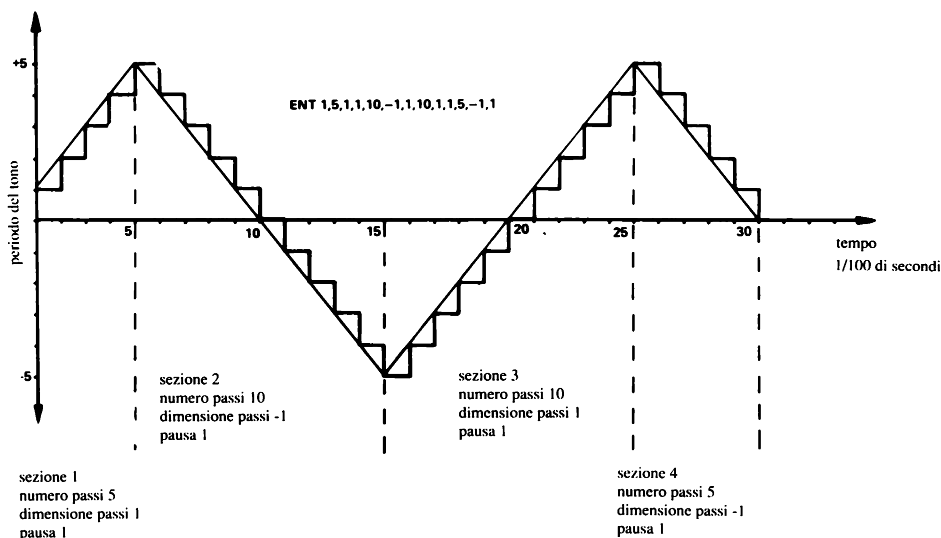


Figura 4: Caratteristiche dell'inviluppo del tono

La principale differenza tra gli inviluppi del volume e gli inviluppi dei toni sta nel fatto che gli inviluppi dei toni non hanno alcun effetto sulla durata della nota definita precedentemente con il comando SOUND o con l'inviluppo del volume. Se l'inviluppo del tono finisce prima della fine della nota, verrà mantenuto il periodo del tono finale. Se invece viene oltrepassata la durata della nota, le parti avanzanti vengono perse.

Per ripetere un inviluppo dei toni per tutta la durata del suono, si usi un numero di inviluppo negativo. Nota: Non viene richiamato in negativo in un comando SOUND.

Forma del comando:

ENT S,T1,V1,W1,T2,V2,W2,T3,V3,W3,T4,V4,W4,T5,V5,W5

S: Numero dell'inviluppo	valori tra 1 e 15
T1 .. T5: Numero passi (sezioni 1 .. 5)	valori tra 0 e 239
V1 .. V5: Dimensione passi (sezioni 1 .. 5)	valori tra -128 e 127
W1 .. W5: Pausa (sezioni 1 .. 5)	valori tra 0 e 255 (1/100 di secondi)

S è obbligatorio ed è necessario specificare completamente ogni sezione.

Quando viene definito un inviluppo, tutti i valori precedenti vengono eliminati. Specificare un inviluppo di qualsiasi tipo senza alcuna sezione riporterà tutti i valori che costituivano l'inviluppo a zero.

6.13 Altri comandi e funzioni associati

SQ(x)

x è il numero del canale e può essere uguale a 1, 2 o 4

Questi valori rappresentano rispettivamente i canali A, B e C come è stato mostrato nella tabella del significato dei bit dello <stato del canale>, un parametro del comando SOUND. In questo modo si richiede lo stato del canale specificato.

Questa è una funzione e la risposta è costituita da un intero compreso tra 0 e 255. Per estrarre l'informazione da questo numero, si può controllare la seguente tabella sul significato dei bit:

Bit 0 .. 2	Numero di spazi liberi nella coda verificata (0 .. 4)
Bit 3	Rendezvous con il canale A in testa alla coda
Bit 4	Rendezvous con il canale B in testa alla coda
Bit 5	Rendezvous con il canale C in testa alla coda
Bit 6	Attendi in testa alla coda
Bit 7	Canale attualmente attivo

I bit da 3 a 6 (rendezvous e attendi) sono stati descritti nel paragrafo dedicato allo stato del canale. La sola altra caratteristica di questa funzione di test è la disabilitazione del comando ON SQ GOSUB descritto nel prossimo paragrafo.

ON SQ GOSUB

ON SQ(y) GOSUB <numero di linea>

y è il numero del canale: 1, 2 o 4

Questo è un comando di interruzione che, una volta attivato controlla quando viene terminata una data coda del comando SOUND; in questo caso ritorna alla subroutine SOUND. Le parole chiave SQ e SOUND disabilitano l'interrupt. La subroutine termina come di consueto con il comando RETURN. Ogni canale ha la stessa priorità e, quando viene trovato uno spazio vuoto nella coda del canale specificato, il comando scatta, disabilitandosi nello stesso tempo. Perciò la subroutine deve ripristinare l'interrupt, se è necessario utilizzarlo nuovamente.

RELEASE

RELEASE z

z indica il o i numeri di canale mediante un intero che può andare da 1 a 7.

Quando si è descritto il parametro <stato del canale> del comando SOUND, si è detto che è possibile far attendere un canale mediante un particolare comando SOUND. Il comando RELEASE permette semplicemente di rimuovere il blocco posto in tal modo. I bit che formano il parametro specificano i canali da far proseguire. Facendo proseguire un canale che non era in attesa, non si ottiene alcun effetto.

Bit 0: Canale A
Bit 1: Canale B
Bit 3: Canale C

7 Stampanti e joystick

Il CPC464 non richiede ulteriori interfacce per controllare due joystick, o una stampante compatibile parallela.

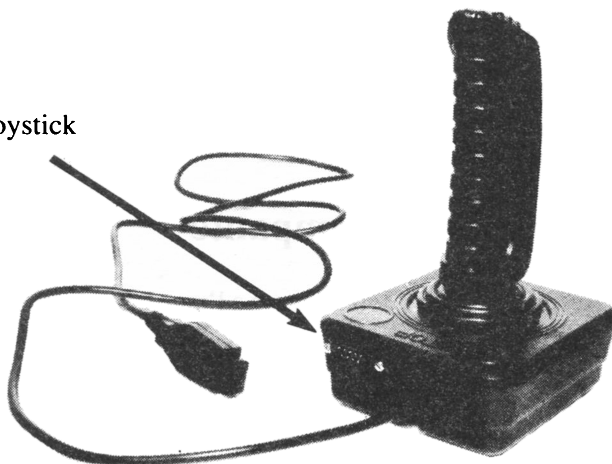
Gli argomenti trattati in questo capitolo sono:

* Joystick
* Stampanti parallele
* Intefacciamento

Il joystick modello JY1 è un oggetto addizionale che si potrebbe voler acquistare per giocare con il CPC 464 e che incorpora la possibilità di controllare e di 'far fuoco' all'interno del gioco. Il JY1 può essere collegato al retro del computer usando la presa **USER PORTS (I/O)**. Il CPC464 Amstrad può operare con due joystick. Il secondo può essere collegato al primo.

Vi sono due cose da comsiderare quando si connette un joystick al CPC464: il JY1 può essere collegato direttamente alla presa 9 pin **USER PORTS (/O)** sul retro del computer. Il secondo joystick può essere connesso alla presa alla base del primo joystick. Le connessioni tra questi dispositivi sono mostrate nell'Appendice V al termine di questo manuale.

Presa per il secondo joystick



joystick JY1

7.1 Il Joystick

Il software del computer può gestire uno o due joystick. Essi vengono gestiti come una parte della tastiera ed in tal modo possono essere interrogati mediante **INKEY** e **INKEY$**. Si noti che nella maggior parte dei casi, il tasto principale "fire" di un joystick viene interpretato come "fire 2" dal 464.

Le funzioni Joy(0) e Joy (1) permettono il controllo rispettivamente del primo e del secondo joystick. La funzione restituisce un risultato mappato bit a bit che indica lo stato degli interruttori del joystick all'ultima scansione della tastiera.

Il joystick restituisce i valori che seguono nella tabella dove KEY è il valore usato dalla funzione INKEY e MIRROR quello equivalente alla tastiera:

Primo Joystick	JOY (0)	KEY	Secondo Joystick	JOY (1)	KEY	MIRROR
Alto	Bit 0	72	Alto	Bit 0	48	6
Basso	Bit 1	73	Basso	Bit 1	49	5
Sinistra	Bit 2	74	Sinistra	Bit 2	50	R
Destra	bit 3	75	Destra	Bit 3	51	T
Fire 2	bit 4	76	Fire 2	Bit 4	52	G
Fire 1	Bit 5	77	Fire 1	Bit 5	53	F

Si noti che quando i valori dei tasti del secondo joystick vengono letti, il computer non può sapere se tali valori sono stati provocati dal joystick o da uno degli equivalenti tasti della tastiera. Ciò significa che la tastiera può essere usata per sostituire il secondo joystick.

Quando si usa il secondo joystick *JY1* questo è identico al primo e non necessitando di una ulteriore presa può essere collegato al primo.

La presa 9 pin **USER PORTS (I/O)** accetta i joystick standard che lavorano con gli altri personal computer sebbene in tal caso non sia possibile collegare a questi il secondo joystick: occorrerà in tal caso usare uno speciale adattatore. Comunque non si consiglia di usare uno di tali joystick da collegare come secondo joystick al *JY1* .

Chi scrive il software potrebbe fornire una opzione all'inizio del programma che abilita l'utente a selezionare le opzioni del joystick o i cursori (dove il tasto **[COPY]** o altri tasti possono essere il pulsante fire).

7.2 Interfacce per la stampante

Il CPC AMSTRAD permette la connessione e l'uso di interfacce per stampanti parallele Centronics.

Il cavo della stampante è stato costruito semplicemente per connettere la porta PRINTER e il connettore della stampante parallela. Si noti che vi sono due chiavi in meno sul circuito stampato del computer e questo permette l'uso di un connetore per circuiti stampati standard.

I dettagli relativi alle interfacce si trovano nell'Appendice V.

Il cavo deve essere costruito in modo che il pin 1 del computer sia connesso con il pin 1 sulla stampante, il pin 19 del computer deve essere connesso al 19 pin della stampante, ecc, con i pin 18 e 36 della stampante non connessi al computer.

In particolare la fila inferiore di pin sul computer è numerata fino a 19 in modo che ogni elemento sia connesso con il pin con lo stesso numero nel connettore della stampante.

Il computer usa il segnale *BUSY* (occupato) (pin 11) per sincronizzarsi con la stampante ed attende se il segnale è *OFF LINE* (Non in linea).

Non sono richiesti comandi utente, e l'output viene diretto alla stampante mediante il canale 8:

```
LIST #8
```

Ciò fa si che il programma residente in memoria venga listato sempre che questo non sia protetto.

All'interno dei programmi la stampante può essere indirizzata usando il comando:

```
PRINT #8,"Questo viene inviato alla stampante"
```

Molte stampanti vanno a capo automaticamente nel caso la linea inviata sia troppo lunga: si controlli sul manuale della propria stampante. Il BASIC AMSTRAD ridirige l'output anche mediante un comando WIDTH (ampiezza). Il valore per difetto della ampiezza della stampante è 132 e può essere modificato in qualsiasi altro: ad esempio WIDTH 80

Se viene specificato il valore 255 allora il BASIC AMSTRAD non invierà i segnali di a capo e delegherà alla stampante il compito di controllare la fine linea. BASIC tiene traccia della posizione della stampa in un contatore: si può conoscere tale valore mediante la funzione POS.

```
IF POS(#8) > 50 THEN GOTO 100
```

Il CPC464 invia un line feed CHR$(10) e un carattere di a capo CHR$(13) al termine di ogni linea. La stampante contiene una commutazione prestabilita per una appropriata forma di input e sarà facile trovare quella più giusta.

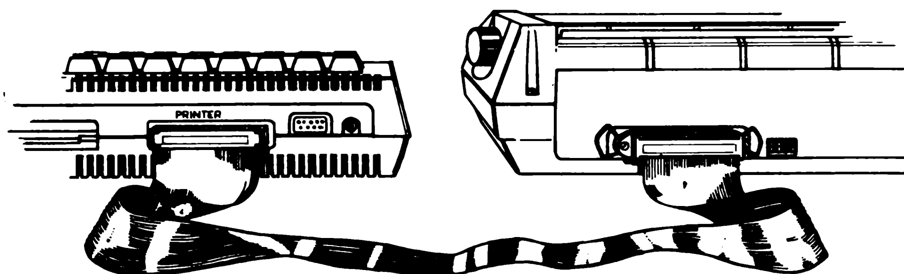
7.3 Stampante grafica

Il manuale fornito con la stampante specifica che i caratteri di controllo sono generalmente nella forma:

```
PRINT CHR$(n)
```

Alcune stampanti hanno caratteri molto simili a molti caratteri grafici AMSTRAD elencati nell'Appendice III ma difficile che i caratteri corrispondano esattamente; si deve quindi consultare la propria tabella di conversione.

Sebbene l'interfaccia permetta di usare stampanti a matrice a basso costo, essa potrà supportare stampanti a margherita dotate della giusta interfaccia, plotter e stampanti a più colori. La chiave di tale compatibilità va ricercata nell'interfaccia parallela standard.



8 Guida sintetica di riferimento al BASIC AMSTRAD

Un elenco di comandi BASIC illustrati da alcuni esempi, che forniscono la sintassi precisa e l'uso delle parole chiave associate ad essi.

Argomenti di questo capitolo

* Convenzioni nelle definizioni
* Caratteri speciali e loro significato
* Tutte le parole chiave del BASIC AMSTRAD in ordine alfabetico

Questo capitolo contiene un sintetico sommario delle funzioni del BASIC contenuto nella ROM del CPC464. Le varie caratteristiche disponibili rappresentano una implementazione standard del linguaggio, con alcune estensioni che permettono di utilizzare le caratteristiche dell'hardware del CPC464.

8.1 Notazione

Caratteri speciali

& o &H	Prefisso di una costante esadecimale
&X	Prefisso di una costante binaria
:	Separa vari comandi immessi nella stessa linea
#	Prefisso di un indicatore di canale

Tipi di dati

Le stringhe possono essere composte da un numero di caratteri compreso tra 0 e 255, e vengono nominate come <espressioni stringa>. Possono essere accodate le une alle altre mediante l'operatore +, considerando che la lunghezza totale non deve comunque superare i 255 caratteri.

I dati numerici possono essere interi o reali. Gli interi permessi sono quelli compresi tra -32768 e 32767 mentre i numeri reali, con nove cifre di precisione possono essere compresi tra +/- 1.7E+38 con il valore più piccolo maggiore di zero circa 2.9E-39.

Gli identificatori di tipo sono % per gli interi, ! per i reali e $ per le stringhe.

Una <espressione numerica> è una qualunque espressione che dà come risultato un numero: può essere costituita semplicemente da un numero, o da una variabile numerica o composta da numeri, operatori e variabili. In nessun caso può essere utilizzata una stringa.

Una <espressione di canale> è costituita da una <espressione numerica> che identifica lo schermo, la stampante o la cassetta dove deve essere inviato il testo specificato.

Se si ottiene un messaggio di "improper argument" (argomento inadatto) significa che una espressione numerica ha restituito un valore che supera i limiti numerici permessi in quella situazione o che un parametro del comando è in qualche modo inadatto e che il BASIC non lo può accettare.

PAROLE CHIAVE

Le parole chiave del BASIC AMSTRAD sono state elencate usando la forma:

PAROLA CHIAVE
Sintassi/funzione
Esempio
Descrizione
PAROLE CHIAVE associate

IMPORTANTE:

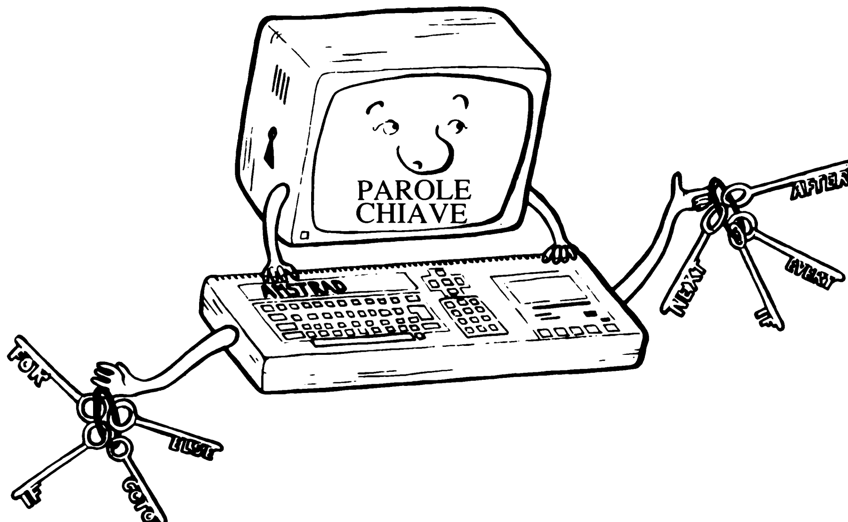
Le parole chiave possono essere
COMANDI: operazioni eseguibili direttamente.
FUNZIONI: operazioni richiamate come argomento all'interno di una espressione

Parentesi
() sono necessarie per l'esecuzione del comando o della funzione.
Altri tipi di parentesi presenti nella descrizione di un comando sono solo descrittive e non devono essere immesse:
[] racchiudono termini opzionali
<> racchiudono espressioni varie che descrivono l'oggetto che sostituiscono

Virgolette
Il BASIC utilizza solo le virgolette doppie ". Le virgolette semplici ' vengono usate solo per evidenziare certi aspetti della descrizione e non appariranno nella specificazione della sintassi-funzione, nè negli esempi, tranne che all'interno delle <espressioni stringa>.

Immissione
Il BASIC converte in maiuscolo tutte le parole chiave immesse in minuscolo. Negli esempi mostrati infatti esse appariranno in maiuscolo in quanto questo il modo in cui il programma appare quando viene LISTato. Se si immetteranno le parole chiave in minuscolo sarà più facile trovare eventuali errori di battitura, in quanto le parole chiave scritte in modo errato appariranno ancora in minuscolo quando il programma verrà LISTato.

Le parole chiave vengono delimitate da separatori e dunque il BASIC AMSTRAD permette di inserire il nome di un comando all'interno del nome di una variabile: ad esempio, lista e continuazione potranno essere nomi di variabili accettabili nel BASIC AMSTRAD.



ABS

ABS(<espressione numerica>)

```
PRINT ABS(-67.98)
 67.98
```

FUNZIONE: Restituisce il valore assoluto di una data espressione - in pratica i numeri negativi vengono cambiati di segno.

Parole chiave associate: SGN

AFTER

AFTER <espressione intera>[,<espressione intera>] GOSUB <numero di linea>

```
AFTER 200,2 GOSUB 320
```

COMANDO: Chiama una subroutine dopo che è passato un certo periodo di tempo. La prima <espressione intera> indica il ritardo, in 1/50 di secondi; la seconda <espressione intera> (compresa tra 0 e 3) indica quale dei 4 timer di ritardo deve essere usato. Vedere anche il Capitolo 10.

Parole chiave associate: EVERY, REMAIN

ASC

ASC(<espressione stringa>)

```
PRINT ASC("X")
 88
```

FUNZIONE: Fornisce il valore numerico nel codice ASCII del primo carattere della stringa fornita.

Parole chiave associate: CHR$

ATN

ATN(<espressione numerica>)

```
PRINT ATN(1)
 0.785398163
```

FUNZIONE: Calcola l'arco-tangente (forzando la <espressione numerica> in un numero reale compreso tra -PI/2 e PI/2) del valore specificato.

Parole chiave associate: SIN, COS, TAN, DEG, RAD

AUTO

AUTO [<numero di linea>][,<incremento>]

```
AUTO 100,50
```

COMANDO: Genera automaticamente i numeri di linea. Il <numero di linea> specifica la prima linea che deve essere generata, in caso la si voglia aggiungere alla fine del programma. Il valore dell'<incremento> specifica la differenza tra due numeri di linea. Per difetto entrambi assumono il valore 10.

Se si rischia di cancellare qualche linea di programma, il BASIC inserirà come avvertimento un asterisco *.

BIN$

BIN$(<espressione intera senza segno>[,<espressione intera>])

```
PRINT BIN$(64,8)
01000000
```

FUNZIONE: Produce una stringa di cifre binarie che rappresentano il valore dell'<espressione intera senza segno>, riempiendo con degli zeri il numero di cifre specificate dalla seconda <espressione intera>.

Parole chiave associate: HEX$, STR$

BORDER

BORDER <colore>[,<colore>]

```
BORDER 3,2
```

COMANDO: Cambia il colore della cornice dello schermo. Se vengono specificati due colori, la cornice alterna i due colori con la frequenza determinata da un eventuale comando SPEED INK. I colori per la cornice possono assumere i valori tra 0 e 26.

Parole chiave associate: SPEED INK

CALL

CALL <espressione di indirizzo>[<lista di parametri>]

CALL &BD19

COMANDO: Permette di chiamare da BASIC una subroutine sviluppata esternamente. E' una istruzione da usare con cautela; non è una che possa essere provata da un utente inesperto. La CALL indicata come esempio è piuttosto innocua, in quanto attende semplicemente il prossimo controllo dello schermo e può essere molto utile se si intendono produrre effetti di animazione.

Parole chiave associate: UNT

CAT

CAT

CAT

COMANDO: Indica al BASIC di scandire tutta la cassetta e di visualizzare i nomi dei file trovati. In questo modo non si altera in alcun modo il programma che si trova in memoria e può essere utilizzato per verificare un programma che è appena stato salvato prima di modificare la memoria del programma. Il comando chiede anche di premere il tasto Play del registratore e, quando trova un programma risponde:

FILENAME BlockNumber Flag Ok

I flag indicano il tipo di registrazione:
$ Un programma BASIC
% Un file di BASIC protetto
* Un file di testo ASCII
& Un file binario

Potrebbero apparire altri caratteri nel caso che il file in questione non sia stato prodotto dal BASIC

Parole chiave associate: LOAD, RUN, SAVE

CHAIN
CHAIN MERGE

CHAIN <nome di file>[,<espressione di numero di linea>]
CHAIN MERGE <nome di file>[,<espressione di numero di linea>]
[,DELETE <numeri di linea>]

CHAIN "PROVA", 350

COMANDO: CHAIN carica un programma dalla cassetta in memoria, sostituendo il corrente programma. CHAIN MERGE unisce il programma contenuto nella cassetta a quello contenuto in memoria. La <espressione di numero di linea> indica il numero di linea da cui deve iniziare l'esecuzione quando il prossimo programma della catena sarà caricato. In assenza della <espressione di numero di linea>, il BASIC inizierà l'esecuzione dal più basso numero di linea.
Se non viene specificato alcun nome di file, il BASIC cercherà di caricare il primo file valido incontrato sul nastro. Se il primo carattere del <nome di file> è !, questo non andrà a comporre il nome del file ed eviterà l'apparire sullo schermo del consueto messaggio di caricamento da cassetta.
CHAIN MERGE mantiene tutte le variabili, anche se le funzioni utente ed i file aperti verranno persi. ON ERROR GOTO viene disattivato, viene effettuato un RESTORE e vengono reinizializzati i valori di DEFINT, DEFREAL e DEFSTR. Vengono dimenticati tutti i FOR, i WHILE ed i GOSUB attivi. Non sarà possibile caricare i file protetti.

Parole chiave associate: LOAD e MERGE

CHR$

CHR$(<espressione intera>)

```
PRINT CHR$(100)
 d
```

FUNZIONE: Converte un valore numerico nell'equivalente carattere (usando il set di caratteri dell'AMSTRAD CPC464 elencati nell'Appendice III)

Parole chiave associate: ASC, LEFT$, RIGHT$, MID$, STR$

CINT

CINT(<espressione numerica>)

```
10 n=578.76543
20 PRINT CINT(n)
RUN
 579
```

FUNZIONE: Converte il valore fornito ad un intero arrotondato compreso tra -32768 e 32767.

Parole chiave associate: CREAL, INT, FIX, ROUND, UNT

CLEAR

CLEAR

CLEAR

COMANDO: Cancella tutte le variabili ed i file.

CLG

CLG [<colore mascherato>]

CLG

COMANDO: Cancella lo schermo grafico.

Parole chiave associate: CLS, ORIGIN

CLOSEIN

CLOSEIN

CLOSEIN

COMANDO: Chiude il file di input dalla cassetta. I comandi NEW e CHAIN MERGE elimineranno tutti i file aperti.

Parole chiave associate: OPENIN, CLOSEOUT

CLOSEOUT

CLOSEOUT

CLOSEOUT

COMANDO: Chiude il file di output sulla cassetta.

Parole chiave associate: OPENOUT, CLOSEIN

CLS

CLS[#<espressione di canale>]

CLS

COMANDO: Cancella lo schermo.

CONT

CONT

CONT

COMANDO: Continua l'esecuzione del programma dopo un *Break*, uno STOP o un END, a meno che il programma sia stato nel frattempo alterato. Può essere immesso come comando diretto.

COS

COS(<espressione numerica>)

```
?COS(34)
-0.848570274
```

e

```
deg:?cos(34)
0.829037573
```

FUNZIONE: Calcola il coseno del valore fornito. La funzione utilizza la misurazione degli angoli in radianti a meno che sia stato specificato altrimenti con il comando DEG. Nell'esempio, il punto di domanda ? viene utilizzato come abbreviazione del comando PRINT e le istruzioni sono state immesse in minuscolo, una possibilità ammessa dal BASIC AMSTRAD .

Parole chiave associate: SIN, TAN, ATN, DEG, RAD

CREAL

CREAL(<espressione numerica>)

```
5 DEFINT n
10 n=75.765
20 d=n/34.6
30 PRINT d
40 PRINT CREAL(n)
50 PRINT n/55.4
run
2.19653179
76
1.37184116
Ready
```

FUNZIONE: Converte un valore in un numero reale.

Parole chiave associate: CINT

DATA

DATA <elenco di costanti>

```
10 REM esempio di lettura con linee DATA
20 DIM Prova$(5)
30 DIM CognomeProva$(5)
40 FOR n=1 TO 5
50 READ Prova$(n)
60 READ CognomeProva$(n)
70 PRINT Prova$(n);" "CognomeProva$(n)
80 DATA Roberto,Bianchi,Dino,Verdi,Piero,Rossi,Aldo,Bruni,Ivo,Risi
90 NEXT
```

COMANDO: Dichiara delle costanti da utilizzare all'interno del programma. E' una delle caratteristiche più usate del BASIC . I tipi di dati devono essere concordi con la variabile in cui verranno memorizzati. Un comando DATA può trovarsi in qualunque punto del programma.

Parole chiave associate: READ, RESTORE

DEF FN

DEF FN <nome>[(<parametri formali>)]=<espressione generale>

```
10 DEF FN interesse(cifra)=1.18*(cifra)
20 INPUT "Somma iniziale",cifra
30 PRINT "La quantità con gli interessi di un anno  di"FNinteresse(cifra)
```

COMANDO: Il BASIC permette di definire all'interno del programma semplici funzioni che calcolano alcuni valori. La parte di definizione è costituita dalle parole DEF FN che creano una funzione specifica per il programma che opererà esattamente come una funzione interna del BASIC come COS.
Può essere chiamata dall'interno del programma. I tipi delle variabili devono concordare ed il comando DEF FN dovrebbe essere scritto all'esterno di un ciclo.

DEFINT
DEFSTR
DEFREAL

DEFtipo <insieme di lettere>

```
DEFINT  I-N
DEFSTR  A,W-Z
DEFREAL
```

COMANDO: Definiscono il tipo di una variabile dove "tipo" può essere intero, reale o stringa. La variabile sarà fissata secondo la prima lettera del suo nome - che può essere in maiuscolo o in minuscolo.

Parole chiave associate: LOAD, RUN, CHAIN, NEW, CLEAR

DEG

DEG

```
DEG
```

COMANDO: Fissa la misurazione degli angoli in gradi. Per difetto la misurazione degli angoli viene effettuata in radianti. Il comando fisserà la misurazione in gradi fino all'intervento di un comando CLEAR o RAD - o fino al caricamento di un altro programma.

Parole chiave associate: RAD

DELETE

DELETE <numeri di linea>

DELETE 100-200

COMANDO: Un comando che cancella dal programma le linee specificate. Se viene scritto per sbaglio, non è più possibile recuperare le linee perdute. E' un comando da usare con cura; si controlli attentamente prima di inviarlo.

Parole chiave associate: NEW

DI

DI

```
10 CLS
20 TAG
30 EVERY 10 GOSUB 100
40 x1=RND*320:x2=RND*320
50 y=200+RND*200
60 FOR x=320-x1 TO 320+x2 STEP 2
70 DI:PLOT 320,0,1:MOVE x-2,y:PRINT " ";:MOVE x,y:PRINT "#";:EI
80 NEXT
90 GOTO 40
100 MOVE 320,0
110 DRAW x+8,y-16,0
120 RETURN
```

COMANDO: Disabilita gli interrupt (interruzioni) tranne che una interruzione *Break*. Le interruzioni vengono abilitate esplicitamente da EI o implicitamente da un RETURN alla fine di una routine di interrupt richiamata con GOSUB.

Viene usato quando il programma deve eseguire certe istruzioni senza alcuna interruzione - ad esempio quando due routine all'interno del programma cercano di ottenere l'uso di una risorsa. Nell'esempio qui sopra il programma principale e la subroutine di interrupt sono in competizione per l'uso dello schermo grafico.

Parole chiave associate: EI

DIM

DIM <lista di variabili con indice>

```
10 CLS:PRINT "Immettere 5 nomi....":PRINT
20 DIM b$(5)
30 FOR n=1 TO 5
40 PRINT "Prego, il nome";n;
50 INPUT b$(n)
60 NEXT
70 FOR n=1 TO 5
80 PRINT "Quanto sei stato saggio ";b$(n);" ad acquistare il CPC464"
90 NEXT
```

COMANDO: Alloca dello spazio per dei vettori e specifica il valore massimo dell'indice. Il BASIC deve sapere quanto spazio deve riservare ad un vettore, o, per difetto, assumerà 10. Una volta fissate, implicitamente o esplicitamente, le dimensioni del vettore, queste non possono essere modificate in caso contrario verrà presentato un messaggio di errore.

Una variabile con indice è una variabile che può assumere una serie di valori come stabilito dal numero intero che determina così la lista delle dimensioni. Il primo numero della lista può essere pensato come il numero delle file di macchine in un parcheggio, il secondo come il numero di macchine che possono stare in una fila, ecc. Una comprensione più approfondita dei vettori è uno degli elementi più importanti per la programmazione avanzata in BASIC. Le dimensioni di un vettore sono limitate solo dalla memoria disponibile e dalla capacità del programmatore di rendersi conto dell'effetto dei vari indici della lista delle dimensioni.

Parole chiave associate: ERASE

DRAW

DRAW <coordinata-x>,<coordinata-y>[,<colore mascherato>]

```
DRAW 200,200,13
```

COMANDO: Disegna sullo schermo una linea, a partire dalla posizione del cursore grafico fino al punto specificato in coordinate assolute. La posizione delle coordinate non viene modificata nelle diverse modalità di schermo. Ulteriori esempio nel Capitolo 5.

Parole chiave associate: DRAWR, PLOT, PLOTR, MOVE, MOVER, TEST, TESTR, XPOS, YPOS, ORIGIN

DRAWR

DRAWR <spostamento-x>,<spostamento-y>[,<colore mascherato>]

```
DRAWR 200,200,13
```

COMANDO: Disegna sullo schermo una linea, a partire dalla posizione del cursore grafico fino al punto specificato in coordinate relative al primo punto.

Parole chiave associate: DRAW, PLOT, PLOTR, MOVE, MOVER, TEST, TESTR, XPOS, YPOS, ORIGIN

EDIT

EDIT <numero di linea>

EDIT 110

COMANDO: Richiama una determinata linea di programma. Vedere il Capitolo 1.4.

Parole chiave associate: LIST

EI

EI

EI

COMANDO: Abilita gli interrupt (interruzioni) disabilitati da DI

Parole chiave associate: DI

END

END

END

COMANDO: Fine del programma. E' una istruzione opzionale, se dopo la fine del programma non vi sono altre istruzioni. END chiude tutti i file aperti sulla cassetta e ritorna al modo diretto. Le code del suono continuano fino allo svuotamento.

Parole chiave associate: STOP

ENT

ENT <numero dell'inviluppo>[,<sezioni dell'inviluppo>]

```
10 ENT 1,100,2,20
20 SOUND 1,100,1000,4,0,1
```

COMANDO: Durante la generazione del suono è possibile cambiare il tono. Un inviluppo del tono definisce il modo in cui il tono deve essere variato. Vedere il Capitolo 6 e l'Appendice VII per una spiegazione approfondita del suono e del suo utilizzo.
Il <numero dell'inviluppo> è una <espressione intera> il cui valore assoluto deve essere compreso tra 1 e 15, e specifica l'inviluppo del tono da modificare. Se l'inviluppo è negativo, esso verrà ripetuto.

Possono essere specificate fino a 5 sezioni dell'inviluppo, ognuna delle quali deve avere la forma:

<numero passi>,<dimensione passi>,<pausa>

o:

<periodo del tono>,<pausa>

La prima forma specifica una modifica incrementale relativa all'attuale periodo del tono. La seconda forma specifica un valore assoluto per il periodo del tono.

<numero passi> fornisce il numero di passi contenuto in quella sezione ed è una <espressione intera> compresa tra 0 e 239

<dimensione passi> fornisce la quantità di cui deve variare il tono per ogni passo dell'inviluppo ed è una <espressione intera> compresa tre -128 e +127.

<pausa> fornisce il tempo che occorre attendere tra i vari passi ed è una <espressione intera> che specifica la durata in unità di 0.01 secondi. L'espressione deve fornire un valore compreso tra 0 e 255 dove 0 viene trattato come fosse 256.

<periodo del tono> fornisce un nuovo valore per il periodo ed è una espressione intera compresa tra 0 e 4095.

Il comando SOUND fissa il periodo iniziale del tono e può specificare uno dei quindici inviluppi del tono. Se non viene specificato alcun inviluppo o se viene specificato un inviluppo che non è stato precedentemente preparato, il tono rimane costante per tutto il suono.

Un inviluppo del tono non ha alcun effetto sulla durata del suono. Se, quando la nota finisce rimangono da eseguire ancora altri passi, questi non vengono eseguiti.

Un inviluppo che si ripete viene fatto ripartire ogni volta che finisce, fino al termine della nota.

Le espressioni dell'inviluppo del tono vengono valutate quando il comando viene eseguito ed i risultati vengono memorizzati per un successivo uso. Quando si usa un inviluppo, non deve essere rieseguito questo comando.

Ogni volta che un dato inviluppo viene fissato, il suo valore precedente viene perso. Se si cambia un inviluppo mentre un suono lo sta utilizzando si potrà produrre effetti indeterminati (anche se interessanti).

Se si specifica un inviluppo senza alcuna sezione, viene cancellato il valore precedentemente memorizzato per quell'inviluppo. Ogni successivo uso di quell'inviluppo verrà ignorato.

Parole chiave associate ENV, SOUND

ENV

ENV <numero dell'inviluppo>[,<sezioni dell'inviluppo>]

```
10 ENV 1,100,2,20
20 SOUND 1,100,1000,4,1
```

COMANDO: Durante la generazione del suono è possibile cambiarne il volume. Un inviluppo del volume è costituito da:

<numero passi>,<dimensione passi>,<pausa>

e definisce il modo in cui il volume deve essere variato specificando un numero di passi che può essere compreso tra 0 e 127, le dimensioni di un passo, comprese tra-128 e +127 e la pausa in intervalli di 0.01 secondi con valori tra 1 e 256.

Il <numero dell'inviluppo> è una <espressione intera> il cui valore assoluto deve essere compreso tra 1 e 15, e specifica l'inviluppo del volume da modificare,

Possono essere specificate fino a 5 sezioni dell'inviluppo, ognuna delle quali deve avere la forma:

<numero passi>,<dimensione passi>,<pausa>

o:

<inviluppo hardware>,<periodo dell'inviluppo>

La prima forma specifica una modifica sotto il controllo del software ed i suoi parametri sono:

<numero passi> fornisce il numero di passi contenuto in quella sezione ed è una <espressione intera> compresa tra 0 e 127

<dimensione passi> è la quantità di cui deve variare il volume per ogni passo dell'inviluppo ed è una <espressione intera> compresa tre -128 e 127.

oppure, se la dimensione passi è zero, il valore a cui fissare il volume è un valore assoluto.

<pausa> il tempo che occorre attendere tra i vari passi ed una <espressione intera> che specifica la durata in unità di 0.01 secondi. L'espressione deve fornire un valore compreso tra 0 e 255 dove 0 viene trattato come fosse 256.

La seconda forma specifica un valore assoluto per l'inviluppo hardware.

<inviluppo hardware> è il valore che deve essere inserito nel registro di forma dell'inviluppo (registro 15 ottale)

<periodo dell'inviluppo> è il valore che deve essere inserito nei registri di periodo dell'inviluppo (registri 13 e 14 ottali)

I valori dell'inviluppo hardware non hanno associata una pausa, perciò, la sezione successiva dell'inviluppo viene eseguita immediatamente. Si raccomanda di dare al passo successivo una pausa di una lunghezza adeguata. Se non vi è alcun passo successivo, viene assunta una pausa di 2 secondi.

Il comando SOUND fissa il volume iniziale e può specificare uno dei quindici inviluppi del volume. Se non viene specificato alcun inviluppo o se viene specificato un inviluppo che non è stato precedentemente preparato, il volume rimane costante per tutto il suono.

Fissando una <dimensione passi> di zero con un numero passi diverso da zero si provoca il mantenimento del volume attuale.

Le espressioni dell'inviluppo del volume vengono valutate quando il comando viene eseguito ed i risultati vengono memorizzati per un successivo uso. Quando si usa un inviluppo, non deve essere rieseguito questo comando.

Ogni volta che un dato inviluppo viene fissato, il suo valore precedente viene perso. Se si cambia un inviluppo mentre un suono lo sta utilizzando si potrà produrre effetti indeterminati (anche se interessanti).

Se si specifica un inviluppo senza alcuna sezione, viene cancellato il valore precedentemente memorizzato per quell'inviluppo. Ogni successivo uso di quell'inviluppo verrà ignorato.

Parole chiave associate: ENT, SOUND

EOF

```
EOF
```

```
PRINT EOF
-1
```

FUNZIONE: Controlla se l'input dalla cassetta ha raggiunto la fine del file. Restituisce -1 (vero) alla fine del file, altrimenti restituisce 0 (falso).

Parole chiave associate: OPENIN

ERASE

ERASE <lista di nomi di variabile>

```
ERASE a, b$
```

COMANDO: Quando un vettore non occorre più, può essere cancellato dalla memoria in modo da usare tale memoria per altri usi.

Parole chiave associate: DIM

ERR
ERL

```
ERR
ERL
```

```
10 CLS
20 ON ERROR GOTO 1000
30 DATA Sandra, Emma, Giovanna, Elena, Gemma
40 READ a$
50 PRINT a$
60 GOTO 40
70 REM il controllo degli errori inizia qui
1000 IF ERR=4 THEN PRINT "Errore: sono finiti i termini"
1010 IF ERL<70 AND ERL>20 THEN PRINT "... nelle linee da 30 a 60"
1020 END
```

VARIABILI: Queste variabili vengono utilizzate nelle sub-routine di gestione degli errori per scoprire un dato errore e la linea in cui è capitato. Si veda l'Appendice VIII.

Parole chiave associate: ON ERROR, ERROR

ERROR

ERROR <espressione intera>

ERROR 17

COMANDO: In caso di un errore, fa una determinata azione. L'errore può essere uno di quelli riconosciuti dal BASIC (Appendice VIII) ed in questo caso l'azione sarà la stessa operata dal BASIC quando rileva tale errore. I numeri di errore dopo quelli riconosciuti dal BASIC possono essere utilizzati per segnalare tipi particolari e personalizzati di errori.

Parole chiave associate: ON ERROR, ERR, ERL

EVERY

EVERY <espressione intera>[,<espressione intera>] GOSUB <numero di linea>
EVERY 500,2 GOSUB 50

COMANDO: Il CPC 464 utilizza un orologio in tempo reale. Questo comando permette di eseguire determinate subroutine ad intervalli regolari. E' possibile utilizzare 4 timer, specificati dalla seconda <espressione intera> con valori da 0 a 3, ad ognuno dei quali è possibile associare una determinata subroutine. Vedere anche il Capitolo 10.

Parole chiave associate: AFTER, REMAIN

EXP

EXP(<espressione numerica>)

```
PRINT EXP(6.876)
 968.743625
```

FUNZIONE: Calcola "E" elevato all'esponente contenuto nella <espressione numerica>, dove "E" è circa 2.7182818 - il numero il cui logaritmo naturale è 1.

Parole chiave associate: LOG

FIX

FIX(<espressione numerica>)

```
PRINT FIX(9.99999)
 9
```

FUNZIONE Diversamente da CINT, FIX elimina semplicemente la parte decimale dell'<espressione numerica> e restituisce la parte intera.

Parole chiave associate: CINT, INT, ROUND

FOR

FOR <variabile> = <inizio> TO <fine> [STEP <passo>]

FOR DAY=1 TO 5 STEP 2

COMANDO: Esegue un comando un determinato numero di volte, eventualmente utilizzando un <passo> per la variabile. Se non viene specificato un <passo> questo viene assunto uguale a 1.

Parole chiave associate: NEXT, WHILE

FRE

FRE(<espressione numerica>)

FRE(<espressione stringa>)

PRINT FRE(0)

PRINT FRE("")

FUNZIONE: Informa sulla memoria rimanente. La forma FRE("") forza una "pulizia della memoria" prima di restituire lo spazio disponibile.

GOSUB

GOSUB <numero di linea>

GOSUB 210

COMANDO: Chiama una subroutine BASIC saltando ad una determinata linea. Vedere RETURN

Parole chiave associate: RETURN

GOTO

GOTO <numero di linea>

GOTO 90

Salta ad un determinato numero di linea.

HEX$

HEX$(<espressione intera senza segno>[,<espressione intera>])

```
PRINT HEX$(65534)
FFFE
```

FUNZIONE: Converte il numero in forma esadecimale. (Vedere Appendice II).
La seconda <espressione intera> può essere utilizzata per specificare la lunghezza minima del risultato.

Parole chiave associate: BIN$, STR$

HIMEM

HIMEM

```
?HIMEM
 43903
```

VARIABILE: Restituisce l'indirizzo dell'ultimo byte di memoria utilizzato dal BASIC e può essere utilizzato nelle espressioni numeriche nel modo usuale.

Prima di ripristinare l'ultimo byte utilizzabile usando il comando MEMORY, conviene utilizzare il comando nn=HIMEM. Sarà in questo modo possibile ritornare alla precedente capacità di memoria eseguendo il comando MEMORY nn.

Parole chiave associate: FRE, MEMORY

IF

IF

IF <espressione logica> THEN <istruzione> [ELSE <istruzione>]

IF <espressione logica> GOTO <numero di linea> [ELSE <istruzione>]

```
IF a>b THEN a=c ELSE a=d

IF a>b GOTO 1000 ELSE 300
```

COMANDO: E' un comando BASIC molto usato per effettuare salti condizionali alle varie parti del programma. La parte logica viene valutata; se vera viene eseguita la parte THEN o GOTO, se falsa, il programma passa all'ELSE o alla linea successiva. E' possibile innestare tra di loro più strutture di questo tipo. L'IF termina alla fine della linea. Non è possibile inserire nella stessa linea altre istruzioni che non siano parte dell'IF.

Parole chiave associate: THEN, ELSE, GOTO, OR, AND, WHILE

INK

INK <inchiostro>, <colore>[,<colore>]

INK 0,1

COMANDO: Vi sono diversi colori disponibili, a seconda della modalità usata per lo schermo (Capitolo 5). I colori usati possono essere modificati con il comando INK, secondo la tabella fornita nell'Appendice IV.

Parole chiave associate: PEN, PAPER

INKEY

INKEY (<espressione intera>)

```
10 CLS:IF INKEY(55)=32 THEN 30 ELSE 20
20 CLS:GOTO 10
30 PRINT "Si è premuto [SHIFT] e V"
40 FOR t=1 TO 1000:NEXT:GOTO 10
```

FUNZIONE: Interroga la tastiera ed indica il tasto premuto. La tastiera viene letta ogni 1/50 di secondo, La funzione è particolarmente utile per ottenere risposte. L'esempio presentato controlla se vengono premuti insieme i tasti **[SHIFT]** e **V**. I tasti **[SHIFT]** e **[CTRL]** vengono identificati secondo lo schema

Valore restituito	**[SHIFT]**	**[CTRL]**	**TASTO**
-1	?	?	ALTO
0	ALTO	ALTO	BASSO
32	BASSO	ALTO	BASSO
128	ALTO	BASSO	BASSO
160	BASSO	BASSO	BASSO

Parole chiave associate: INPUT, INKEY$

INKEY$

INKEY$

```
10 CLS
20 PRINT "Sei intelligente? (s o n) "
30 a$=INKEY$: :IF a$="" THEN GOTO 30
40 IF a$="N" OR a$="n" THEN PRINT "Avresti dovuto acquistarmi!":END
50 IF a$="S" OR a$="s" THEN PRINT "Come sei modesto!!!":END
60 GOTO 20
```

FUNZIONE: Interroga la tastiera per evitare di premere il tasto **[ENTER]** dopo ogni risposta. Se viene premuto un tasto, la funzione restituisce un carattere; se non viene premuto alcun tasto continua a restituire una stringa vuota che viene utilizzata per creare un ciclo che si ripete fino alla pressione di un tasto.

Parole chiave associate: INPUT, INKEY

INP

INP(<porta>)

PRINT INP(&FF77)

FUNZIONE:Restituisce il valore in ingresso dalla porta di I/O specificata nell'indirizzo.

Parole chiave associate OUT, WAIT

INPUT

INPUT [#<canale>,][;][<stringa quotata>;]<lista di variabili>
o:
INPUT [#<canale>,][;][<stringa quotata>,]<lista di variabili>

```
10 CLS
20 INPUT "Scrivi due numeri, separati da una virgola";a,b
30 IF a=b THEN PRINT "I due numeri sono uguali"
40 IF a>b THEN PRINT a " maggiore di" b
50 IF a<b THEN PRINT a " minore di" b
60 CLEAR:GOTO 20
```

COMANDO: Accetta dati in ingresso dal canale specificato.

Il primo punto e virgola è opzionale, elimina il carattere di ritorno carrello che viene altrimenti inserito dopo l'esecuzione del comando. Un punto e virgola provoca la visualizzazione di un punto di domanda; una virgola elimina il punto di domanda. Se viene inserito un dato di tipo errato, come la lettera O invece di 0 (zero), quando si digitano variabili numeriche, BASIC risponde con:

?Redo from start

e visualizza quindi il prompt previsto.

Tutte le risposte date da tastiera devono essere concluse premendo il tasto **[ENTER]**. Il punto e virgola che segue il <canale> elimina il carattere di ritorno carrello che viene altrimenti inserito insieme alla linea lasciando il cursore alla fine del testo immesso. Se si utilizza un canale su cassetta, non viene generato alcun prompt. Se ne viene specificato uno, questo verrà ignorato dal software di gestione della cassetta e dunque il programma può leggere da entrambi i canali.
Verrà letta dal canale un oggetto per ogni variabile specificata. Vi deve essere compatibilità di tipi. Virgole o **[ENTER]** inviati dopo lo spazio saranno ignorati. Le stringhe quotate verranno lette fino alle prossime doppie virgolette, ciò che si trova dopo gli elementi che vengono inseriti viene ignorato. Stringhe non quotate vengono terminate come nel caso dei valori numerici.

Parole chiave associate: LINE INPUT, READ, INKEY$

INSTR

INSTR([<espressione intera>,]<espressione stringa>,<espressione stringa>)

```
PRINT INSTR(2,"BANANA","AN")
```

FUNZIONE: Esamina la prima <espressione stringa> per trovare la seconda <espressione stringa> e restituisce la posizione della sua prima occorrenza. La posizione da cui iniziare la ricerca della stringa può essere specificata mediante il parametro opzionale - altrimenti la ricerca inizia dal primo carattere.

Parole chiave associate: MID$, LEFT$, RIGHT$

INT

INT(<espressione numerica>)

```
PRINT INT(-1.995)
-2
```

FUNZIONE: Arrotonda il numero al minore INTero vicino, eliminando la parte frazionaria. Nel caso di valori positivi restituisce lo stesso valore di FIX, mentre per i valori negativi non interi restituisce un numero in meno rispetto a FIX.

Parole chiave associate: CINT, FIX, ROUND.

JOY

JOY(<espressione intera>)

```
30  IF JOY(0) AND 8 THEN GOTO 100
```

FUNZIONE: Legge un risultato del bit significativo dal JOYstick specificato nell'<espressione intera> (0 o 1).

Bit	Decimale
0:Su	1
1:Giù	2
2:Sinistra	4
3:Destra	8
4:Fuoco 2	16
5:Fuoco 1	32

Parole chiave associate: INKEY.

KEY

KEY <espressione intera>,<espressione stringa>

KEY 140,"RUN"+CHR$(13)

COMANDO: Definisce un nuovo tasto funzione. Possono essere gestiti trentadue simboli di espansione, corrispondenti ai valori da 128 a 159 elencati nell'Appendice III. Quando viene letto uno di questi caratteri, esso viene espanso nella stringa associata. Nell'<espressione stringa> può essere espanso un totale di 100 caratteri.

Parole chiave associate: KEY DEF.

KEY DEF

KEY DEF <numero tasto>,<ripetizione>[,<normale>,[<maiuscolo>[,<control>]]]

KEY DEF 46,1,63

COMANDO: Associa il valore definito nell'Appendice III ad un tasto. L'esempio converte il tasto **N** in modo che stampi un punto interrogativo (il carattere n.63).

Per ritornare alle normali funzioni del tasto:

KEY DEF 46,1,110

dove il carattere 110 è la n minuscola

Parole chiave associate: KEY

LEFT$

LEFT$(<espressione stringa>,<espressione intera>)

```
10 CLS
20 a$="AMSTRAD"
30 b$=LEFT$(a$,3)
40 PRINT b$
run
[cancellazione schermo]
AMS
Ready
```

FUNZIONE: Restituisce un certo numero di caratteri a partire da quello specificato dall'espressione intera dopo averli estratti dalla sinistra (LEFT) dell'<espressione stringa>. Se l'<espressione stringa> è minore della <lunghezza richiesta>, viene restituita tutta l'<espressione stringa>.

Parole chiave associate: MID$, RIGHT$.

LEN

LEN(<espressione stringa>)

```
a$="AMSTRAD":PRINT LEN(a$)
7
```

FUNZIONE: Restituisce il numero totale di caratteri (la lunghezza) dell'<espressione stringa>.

LET

LET <variabile>=<espressione>

```
LET x=100
```

COMANDO: E' un residuo dei primi BASIC in cui si dovevano vedere gli assegnamenti alle variabili. Non viene utilizzato nel BASIC AMSTRAD, e serve solo per assicurare la compatibilità con i programmi contenuti nei manuali dei BASIC più vecchi. L'esempio precedente può essere trasformato semplicemente in:

```
x=100
```

utilizzando il BASIC AMSTRAD

LINE INPUT

LINE INPUT [#<canale>,][;][<stringa quotata>,]<variabile stringa>
LINE INPUT [#<canale>,][;][<stringa quotata>;]<variabile stringa>

```
LINE INPUT a$
LINE INPUT "Nome";n$
```

COMANDO: Accetta in ingresso un'intera linea di testo dal canale indicato. Il primo punto e virgola [;] opzionale elimina il carattere di ritorno carrello/a capo che viene altrimenti inserito dopo l'esecuzione del comando. Il canale normalmente usato è il #0, lo schermo.

Parole chiave associate: READ, INPUT, INKEY$, INKEY.

LIST

LIST [<numeri di linea>][,#<canale>]

```
LIST 100-1000,#1
```

COMANDO: Elenca linee di programma sul canale specificato. Il canale #0 è il canale di schermo, assunto per difetto, mentre #8 è la stampante. L'operazione può essere interrotta temporaneamente premendo una volta **[ESC]** e ripristinata con la pressione della barra spaziatrice. Premendo **[ESC]** due volte si conclude l'operazione, ritornando al modo diretto. I programmi vengono listati per aiutare nella correzione. Il primo o l'ultimo numero del parametro <numeri di linea> possono essere omessi per indicare "dall'inizio del programma" o "fino alla fine del programma", cioè:

```
LIST -200
```

oppure

```
LIST 30-
```

LOAD

LOAD <nomefile>[,<espressione di indirizzo>]

```
LOAD "INVENT"
```

COMANDO: Carica un programma BASIC da cassetta in memoria, sostituendo i programmi eventualmente presenti. Specificando l'<espressione di indirizzo> opzionale si otterrà il caricamento del file binario a quell'indirizzo. Vedere il Capitolo 2.

LOCATE

LOCATE [#<canale>,]<coordinata x>,<coordinata y>

```
10  MODE 1
20 LOCATE 20,12
30  PRINT CHR$(249)
```

COMANDO: Posiziona il cursore sul testo nel canale indicato, alla posizione specificata dalle coordinate x e y, relative all'origine del canale (finestra). Il canale #0 è il canale per difetto.

Parole chiave associate: WINDOW.

LOG

LOG(<espressione numerica>)

```
PRINT LOG(9999)
 9.12024037
```

FUNZIONE: Calcola il logaritmo naturale dell'<espressione numerica>.

Parole chiave associate: EXP, LOG10

LOG10

LOG10(<espressione numerica>)

```
PRINT LOG10(9999)
 3.99995657
```

FUNZIONE: Calcola il logaritmo in base 10 dell'<espressione numerica>.

Parole chiave associate: EXP, LOG.

LOWER$

LOWER$(<espressione stringa>)

```
a$="AMSTRAD": PRINT LOWER$(a$)
amstrad
```

FUNZIONE: Restituisce una nuova espressione stringa che risulta essere una copia dell'<espressione stringa> specificata, nella quale tutti i caratteri alfabetici maiuscoli vengono convertiti in minuscolo. La funzione risulta utile per elaborare dati in ingresso che possono essere in caratteri maiuscoli o minuscoli.

Parole chiave associate: UPPER$.

MAX

MAX(<lista di:<espressione numerica>)

```
10  n=66
20  PRINT MAX(1,n,3,6,4,3)
run
 66
```

FUNZIONE: Restituisce il valore massimo della <lista di:<espressione numerica>.

Parole chiave associate: MIN.

MEMORY

MEMORY <espressione di indirizzo>

```
MEMORY &20AA
```

COMANDO: Ripristina i parametri della memoria del BASIC per cambiare la quantità di memoria disponibile assegnando l'indirizzo del byte più elevato.

Si veda la descrizione del comando HIMEM. Per sapere la quantità di memoria disponibile, usare il comando FRE.

Parole chiave associate: FRE, HIMEM

MERGE

MERGE <nomefile>

MERGE "PROGETTO"

COMANDO: Carica un programma da cassetta e lo aggiunge al programma attualmente in memoria. Se non viene specificato alcun nome di file, il BASIC caricherà il primo programma che incontra sulla cassetta. Se il primo carattere del nome del file è "!", questo verrà eliminato dal nome del file e non verrà presentato sullo schermo il messaggio di caricamento da cassetta. Se si vuole effettuare un MERGE senza sovra-scrivere il programma che si trova attualmente in memoria, è possibile utilizzare il comando RENUM per spostare il programma in una posizione in cui non verrà sovrascritto dal programma caricato. Tutte le variabili, le funzioni utente ed i file aperti verranno cancellati. ON ERROR GOTO viene disabilitato, viene eseguito un RESTORE e vengono reinizializzati DEFINT, DEFREAL e DEFSTR. I file protetti NON possono essere inseriti con MERGE nel programma corrente.

Parole chiave associate: CHAIN, CHAIN MERGE, LOAD.

MID$

MID$(<espressione stringa>,<espressione intera>[,<espressione intera>])

```
a$="AMSTRAD":PRINT MID$(a$,2,4)
MSTR
a$="AMSTRAD":b$=MID$(a$,2,4):PRINT b$
MSTR
```

COMANDO e FUNZIONE: Restituisce una parte della stringa (una sottostringa) che può essere usata come destinazione di un assegnamento (come un comando) o come argomento di una espressione stringa (come una funzione). La prima <espressione intera> è la posizione del primo carattere della sottostringa. Il secondo intero specifica la lunghezza della sottostringa. Se non viene specificato, viene restituito ciò che rimane della <espressione stringa> dopo la posizione iniziale.

Parole chiave associate: LEFT$, RIGHT$.

MIN

MIN(<lista di:<espressione numerica>)

```
PRINT MIN(3,6,2.999,8,9,)
 2.999
```

FUNZIONE: Restituisce il valore minimo della <lista di:<espressione numerica>.

Parole chiave associate: MAX.

MODE

MODE <espressione intera>

MODE 1

COMANDO: Modifica il modo schermo (0,1 o 2) e pulisce lo schermo con il colore 0 (che può non essere il corrente inchiostro della carta). Tutte le finestre di testo e grafica vengono riportate a schermo intero, e i cursori grafico e di testo vengono diretti alle loro rispettive origini.

Parole chiave associate: ORIGIN, WINDOW.

MOVE

MOVE <coordinata x>,<coordinata y>

MOVE 34,34

COMANDO: Sposta il cursore grafico sul punto assoluto specificato dalle <coordinata x> e <coordinata y>. Le corrispondenti funzioni che forniscono la posizione del cursore sono XPOS e YPOS

Parole chiave associate: MOVER, PLOT, PLOTR, DRAW, DRAWR, ORIGIN, TEST, TESTR, XPOS, YPOS.

MOVER

MOVER <coordinata x>,<coordinata y>

MOVER 34,34

COMANDO: Sposta il cursore grafico sul punto relativo specificato dalle <coordinata x> e <coordinata y>. Le corrispondenti funzioni che forniscono la posizione del cursore sono XPOS e YPOS
Parole chiave associate: MOVE, PLOT, PLOTR, DRAW, DRAWR, ORIGIN, TEST, TESTR, XPOS, YPOS.

NEW

NEW

NEW

COMANDO: Cancella il programma corrente e le variabili presenti in memoria. Le definizioni dei tasti non vengono perse e lo schermo non viene pulito.

NEXT

NEXT [<lista di:<variabile>]

FOR n=1 TO 1000:NEXT

COMANDO: Rappresenta il termine di un ciclo FOR. Il comando NEXT può essere anonimo oppure riferirsi al FOR corrispondente.
Nell'esempio precedente si sarebbe potuto scrivere:

NEXT n

Parole chiave associate: FOR

ON GOSUB
ON GOTO

ON <espressione intera> GOSUB <lista di numeri di linea>
ON <espressione intera> GOTO <lista di numeri di linea>

```
10 ON DAY GOSUB 100,200, 300,400, 500
10 ON RATE GOTO 1000,2000,3000,4000
```

COMANDO: Effettua un GOSUB alla subroutine o un GOTO alla istruzione secondo il risultato della <espressione intera>. Se il risultato è 1 viene scelta la prima linea della lista, se è 2 la seconda, e così via. Nella prima linea di esempio quando DAY è uguale a 1 viene eseguita la subroutine alla linea 100. Con DAY uguale a 2 si salta alla linea 200, e così via.

Parole chiave associate: GOTO, GOSUB

ON BREAK GOSUB

ON BREAK GOSUB <numero linea>

```
10  ON BREAK GOSUB 40
20  PRINT "programma in esecuzione"
30  GOTO 20
40  CLS
50  PRINT "premendo due volte [ESC] si richiama la routine GOSUB"
60  FOR t=1 TO 2000:NEXT
70  RETURN
run
```

COMANDO: Ordina a BASIC di passare alla subroutine specificata nel <numero linea> quando il tasto **[ESC]** viene premuto due volte.

Parole chiave associate: ON BREAK STOP, RETURN.

ON BREAK STOP

ON BREAK STOP

```
10 ON BREAK GOSUB 40
20  PRINT "programma in esecuzione"
30 GOTO 20
40 CLS
50 PRINT "hai premuto [ESC] ";
60 PRINT "due volte richiama la routine GOSUB"
70 FOR t=1 TO 2000:NEXT
75 ON BREAK STOP
80 RETURN
run
```

COMANDO: Quando viene eseguito all'interno di una subroutine ON BREAK, ON BREAK STOP disabilita la trappola ma non ha altri effetti immediati. Nell'esempio precedente, il comando ON BREAK GOSUB avrà effetto una sola volta, poichè viene disabilitato dalla linea 75 nella subroutine ON BREAK.

Parole chiave associate: ON BREAK GOSUB.

ON ERROR GOTO

ON ERROR GOTO <numero linea>

```
10  ON ERROR GOTO 80
20  CLS
30  PRINT "Se incontri errori, ";
40  PRINT "lista il programma"
50  PRINT "così puoi trovare l'errore"
60  FOR t=1 TO 4000:NEXT
70  GOSUB 200
80  CLS: PRINT "Errore rilevato alla linea";ERL:PRINT
90  LIST
```

COMANDO: Quando nel programma viene rilevato un errore, si raggiunge il <numero linea> specificato. In questo esempio verrà trovato un errore alla linea 70.

Parole chiave associate: ERL, ERR, RESUME.

ON SQ GOSUB

ON SQ (<canale>) GOSUB <numero di linea>

ON SQ (2) GOSUB 2000

COMANDO: Abilita un interrupt quando vi è uno spazio vuoto nella coda del suono specificata. Il <canale> è una espressione intera contenente uno dei valori:

1: per il canale A
2: per il canale B
4: per il canale C

Parole chiave associate: SOUND, SQ.

OPENIN

OPENIN <nomefile>

COMANDO: Apre (OPEN) un file in ingresso da cassetta, che contiene informazioni che devono venir usate dal programma attualmente nella memoria del computer.

Se il primo carattere di <nomefile> è !, i messaggi relativi alla gestione delle cassette non verranno visualizzati. Il programma caricherà il primo blocco di informazioni da cassetta, pronti per una successiva elaborazione.

Parole chiave associate: CLOSEIN, OPENOUT.

OPENOUT

OPENOUT <nomefile>

COMANDO: Apre un file di output su cassetta, che conterrà informazioni che devono venir usate dal programma attualmente nella memoria del computer. Se il primo carattere di <nomefile> è !, i messaggi relativi alla gestione delle cassette non verranno visualizzati. Il programma creerà il primo blocco di dati in un file con il nome dato. Ogni blocco consiste di 2048 byte di dati.

NB Il comando NEW abbandonerà tutti i file aperti e bufferizzati, il cui contenuto andrà perduto.

Parole chiave associate: CLOSEOUT, OPENIN.

ORIGIN

ORIGIN <x>,<y>[,<sinistra>,<destra>,<alto>,<basso>]

```
10 CLS:BORDER 13
20 ORIGIN 0,0,50,590,350,50
30 DRAW 540,350
40 GOTO 20
```

COMANDO: Determina la posizione iniziale del cursore grafico. La [parte opzionale] del comando contiene le istruzioni per predisporre una nuova finestra grafica, che potrà essere utilizzata in tutti i modi grafici a causa della tecnica di indirizzamento dei pixel che è stata utilizzata.

L'ORIGINe è il punto di coordinate 0,0 (le coordinate crescono verso l'alto e verso destra). Se viene specificata una finestra i cui angoli si trovano all'esterno dello schermo, si assumerà che essi specifichino l'ultima posizione "visibile" all'interno dello schermo.

Parole chiave associate: WINDOW.

OUT

OUT <numero porta>,<espressione intera>

```
OUT &F8F4,10
```

COMANDO: Invia il valore dell'<espressione intera> (compresa tra 0 e 255) alla porta specificata mediante l'indirizzo nel <numero porta>.

Parole chiave associate: INP, WAIT.

PAPER

PAPER [#<canale>],<inchiostro mascherato>

```
10  MODE 0
20 FOR p=0 TO 15
30 PAPER p:CLS
40 PEN 15-p
50 LOCATE 6,12:PRINT "sfondo"p
60 FOR t=1 TO 500;NEXT t
70 NEXT p
```

COMANDO: Assegna il colore dello sfondo per i caratteri. Quando i caratteri vengono scritti sullo schermo di testo, la cella del carattere viene riempita con l'<inchiostro> dello sfondo prima che il carattere venga scritto (se non è stato selezionato il modo trasparente).

Per le correlazioni tra PAPER, MODE e colori, vedere la tabella 2 a pagina F3.4.

Parole chiave associate: INK, WINDOW e PEN.

PEEK

PEEK(<espressione di indirizzo>)

```
10  MODE 2
20  INK 1,0: INK 0,12 : BORDER 12
30  INPUT "Indirizzo iniziale esame memoria";inizio
40  INPUT "Indirizzo finale esame memoria";fine
50  FOR n=inizio TO fine
60  valore$=HEX$(PEEK(n),2)
70  PRINT valore$;
80  PRINT  " pos. ";HEX$(n,4),
90  NEXT
```

FUNZIONE: Esamina il contenuto di una locazione di memoria. Il programma presentato permette di visualizzare il contenuto della RAM del CPC464. Permette di leggere il contenuto della RAM che si trova tra le ROM in posizione &0000-&3FFF e &C000-&FFFF ma non le ROM stesse.

Parole chiave associate: POKE.

PEN

PEN [#<canale>,]<inchiostro>

PEN 1,2

COMANDO: Assegna il valore dell'<inchiostro> che dovrà essere utilizzato quando si disegna su una data finestra; la finestra per difetto è la #0.

Parole chiave associate: INK, PAPER.

PI

PI

```
PRINT PI
3.14159265
```

```
10 REM Disegni in prospettiva
20 MODE 2
30 RAD
40 INK 1,0
50 INK 0,12
60 BORDER 9
70 FOR n=1 to 200
80 ORIGIN 420,0
90 DRAW 0,200
100 REM disegna angoli da un punto che poi sparisce
110 DRAW 30*n*SIN(n*PI/4),(SIN(PI/2))*n*SIN(n)
120 NEXT
130 MOVE 0,200
140 DRAWR 0,50
150 DRAWR 40,0
160 WINDOW 1,40,1,10
170 PRINT "Ora si potrà completare il programma Hangman"
```

FUNZIONE: Restituisce il valore del rapporto tra la circonferenza e il suo diametro. Viene utilizzato molto spesso in routine grafiche come quella mostrata qui sopra.

Parole chiave associate: DEG, RAD.

PLOT

PLOT <coordinata x>,<coordinata y>[,<inchiostro>]

```
10 MODE 2:PRINT "Immetti 4 numeri,
separati da virgole":PRINT
20 PRINT "Origine X (0-639),
Origine Y (0-339), raggio
ed angolo (per il passo)":INPUT x,y,r,a
30 ORIGIN x,y
40 FOR angolo=1 TO 360 STEP a
50 puntox=r*COS(angolo)
60 puntoy=r*SIN(angolo)
70 PLOT puntox,puntoy
80 REM MOVE 0,0
90 REM DRAW puntox,puntoy
100 NEXT
```

COMANDO: Si provi con 320,200,20,1. PLOT si comporta come MOVE ma il pixel di destinazione viene scritto sullo schermo. Se si elimina il comando REM nella linea 90 e si inserisce un REM nella linea 70, se ne inibirà il funzionamento e sarà possibile capire la differenza tra PLOT e DRAW. Si tolga il REM alla linea 80 per riempire il cerchio. Si noti che il programma indica la circonferenza passandovi ripetutamente sopra. Questo programma non ha inizializzato il modo RADianti del calcolo angolare, quindi ogni unità di angolo è notevolmente superiore ad un grado. Si immetta la linea 25 DEG e si esegua nuovamente il programma.

Parole chiave associate: DRAW, DRAWR, PLOT, PLOTR, MOVE, MOVER, ORIGIN, TEST, TESTR, XPOS, YPOS

PLOTR

PLOTR <coordinata x>,<coordinata y>[,<inchiostro>]

```
5 DEG
10 MODE 2:PRINT "Immetti 4 numeri,
separati da virgole":PRINT
20 PRINT "Origine X (0-639),
Origine Y (0-339), raggio
ed angolo (per il passo)":INPUT x,y,r,a
30 ORIGIN x,y
40 FOR angolo=1 TO 360 STEP a
50 puntox=r*COS(angolo)
60 puntoy=r*SIN(angolo)
70 PLOTR puntox,puntoy
80 NEXT : GOTO 40
```

COMANDO: Si provi con 320,0,2,1. PLOTR si comporta come DRAWR ma solo il pixel di destinazione viene scritto sullo schermo.

Parole chiave associate: DRAW, DRAWR, PLOT, PLOTR, MOVE, MOVER, ORIGIN, TEST, TESTR, XPOS, YPOS

POKE

POKE <espressione di indirizzo>,<espressione intera>

POKE &00FF,10

COMANDO: Permette di accedere direttamente alla memoria della macchina ed inserisce una espressione intera compresa tra 0 e 255 nell'indirizzo specificato. Non è un comando da usare con leggerezza.

Parole chiave associate: PEEK.

POS

POS(#<canale>)

```
PRINT POS(#0)
 1
```

FUNZIONE: Restituisce la corrente POSizione all'interno del canale. L'espressione <canale> DEVE essere specificata, ed omettendola si otterrà un errore di sintassi "Syntax error".
Schermo: Restituisce la coordinata X del cursore del testo, relativa all'origine. L'angolo in alto a sinistra viene rappresentato con 1,1.
Stampante: Restituisce la posizione della testina di stampa, dove 1 è il margine sinistro. A tal fine vengono contati tutti i caratteri ASCII con codice maggiore di &1F (31).
Cassetta: Come per la stampante.

Parole chiave associate: VPOS

PRINT

PRINT[#<canale>,][<lista di stampa>][<parte USING>][<separatore>]

PRINT #0,"abc"

COMANDO: Per una spiegazione completa di questo comando, si veda a pagina 54 di questo capitolo.

RAD

RAD

RAD

COMANDO: Attiva il modo di calcolo in RADianti.

Parole chiave associate: ATN, COS, DEG, SIN, TAN.

RANDOMIZE

RANDOMIZE[<espressione numerica>]

```
10 RANDOMIZE 23
20 PRINT RND(6)
```

COMANDO: Il generatore BASIC di numeri casuali produce una sequenza pseudo-casuale nella quale ogni numero dipende dal numero precedente; partendo dal seme specificato, la sequenza ottenuta è sempre la stessa. RANDOMIZE assegna il nuovo valore iniziale per il generatore di numeri casuali sia al valore specificato, sia ad un valore introdotto dall'utente. RANDOMIZE TIME produce una sequenza difficile da ripetere.

Parole chiave associate: RND.

READ

READ lista di:<variabile>

```
10 FOR n=1 TO 4
20 READ a$
30 PRINT a$
40 DATA Paolo, Luca, Davide, Marco
50 NEXT
```

COMANDO: Legge dati dalle istruzioni DATA e li assegna a variabili, spostando quindi automaticamente il puntatore al successivo elemento dell'istruzione DATA. Il comando RESTORE può essere utilizzato per riportare il puntatore all'inizio di una istruzione DATA. Vedere la parola chiave DATA.

Parole chiave associate: DATA, RESTORE.

RELEASE

RELEASE <canali di suono>

RELEASE 4

COMANDO: Quando un suono viene inserito in una coda, può includere uno stato di attesa. Se uno qualunque dei canali specificati in questo canale sono in stato di attesa, verrà rilasciato. L'espressione che identifica il canale dei suoni deve essere letta bit a bit: A=bit 0, B=bit 1, C=bit 2. Perciò 4 (in binario 0100) rilascerà la Coda C.

Parole chiave associate: SOUND.

REM

REM <resto della linea>

```
10  REM Intergalactic Hyperspace Mega-Monster
      Invaders Deathchase
20  REM Copyright AMSTRAD 1985
```

COMANDO: Inserisce un commento in un programma. Il resto della linea viene ignorato da BASIC e può contenere qualsiasi carattere, compresi i due punti : che generalmente separano le istruzioni. Al posto di :REM può essere usato un carattere di apice singolo ' MA NON all'interno di istruzioni DATA, in cui ' viene trattato come parte di una stringa senza apici.

REMAIN

REMAIN(<espressione intera>)

```
REMAIN(3)
PRINT#6, REMAIN(0);
```

FUNZIONE: Disabilita il timer di ritardo specificato (valore compreso tra 0 e 3). Legge dal timer di ritardo il tempo rimanente. Viene restituito zero se il timer non era abilitato.

Parole chiave associate: AFTER, EVERY.

RENUM

RENUM[<nuovo numero linea>][,[<vecchio numero linea>][,<incremento>]]

RENUM

RENUM 100,,100

COMANDO: Rinumera le linee di un programma a partire dalla linea specificata, usando l'incremento specificato. Il <nuovo numero di linea> indica il primo numero per la nuova sequenza, il cui valore per difetto è 10. Il parametro <vecchio numero linea> specifica il numero di linea corrente esistente in cui deve iniziare la rinumerazione. Se questo parametro viene omesso, la rinumerazione inizia dalla prima linea del programma. Il parametro <incremento> specifica il passo numerico tra ciascuna linea rinumerata. Se viene omesso, il valore del passo sarà 10.

RENUM prende in considerazione tutte le chiamate di linea GOSUB, GOTO e altre. Se vengono omessi tutti gli specificatori, verrà eseguito un RENUM 10,,10.
I numeri di linea sono validi se compresi tra 1 e 65535.

RESTORE

RESTORE[<numero linea>]

RESTORE 300

```
10 FOR n=1 TO 7
20 READ a$
30 PRINT a$;" ";
40 DATA i, dati, RESTOREati, possono, essere, letti, ancora
50 NEXT
60 PRINT
70 RESTORE
80 GOTO 10
```

COMANDO: Ripristina la posizione del "puntatore" all'inizio dell'istruzione DATA specificata nel <numero linea> opzionale. Se il parametro viene omesso, il puntatore viene riposizionato all'inizio della prima istruzione DATA.

Parole chiave associate: DATA, READ.

RESUME

RESUME[<numero linea>]

o

RESUME NEXT

RESUME 300

COMANDO: Riprende la normale esecuzione di un programma da un dato numero di linea dopo la rilevazione di un errore trattato mediante un comando ON ERROR GOTO. Se il numero di linea da cui riprendere non è specificato, il programma riprende l'esecuzione dalla stessa linea in cui è stato inizialmente rilevato l'errore. RESUME NEXT porta alla linea immediatamente successiva a quella in cui è stato trovato l'errore.

Parole chiave associate: ON ERROR GOTO.

RETURN

RETURN

COMANDO: Segnala la fine di una subroutine. BASIC rientra dalla subroutine all'istruzione immediatamente successiva al comando GOSUB che l'ha richiamata.

Parole chiave associate: GOSUB, ON x GOSUB, ON SQ GOSUB, AFTER n GOSUB, EVERY n GOSUB, ON BREAK GOSUB

RIGHT$

RIGHT$(<espressione di stringa>,<espressione intera>)

```
10 CLS
20 a$="AMSTRAD"
30 b$=RIGHT$(a$,3)
40 PRINT b$
run
```

[cancellazione schermo]

```
RAD
Ready
```

FUNZIONE: Restituisce il numero di caratteri (compresi tra 0 e 255) specificati nel parametro <espressione intera>, dopo averli estratti dalla destra (RIGHT) dell'<espressione di stringa>. Se l'<espressione di stringa> è più breve rispetto alla <lunghezza richiesta>, viene restituita l'intera <espressione di stringa>.

Parole chiave associate: LEFT$, MID$.

RND

RND[(<espressione numerica>)]

```
10 RANDOMIZE 23
20 PRINT RND
```

FUNZIONE: Restituisce il successivo numero casuale che può essere il successivo nella sequenza, una ripetizione dell'ultimo o il primo di una nuova sequenza. Il comando RANDOMIZE inserito in questo programma, assicura che RND restituisca sempre lo stesso numero. Se l'<espressione numerica> presenta valore zero, RND restituisce una copia dell'ultimo numero casuale generato. Se l'<espressione numerica> presenta valore negativo, la sequenza di numeri casuali è prevedibile.

Parole chiave associate: RANDOMIZE.

ROUND

ROUND (<espressione numerica>[,<espressione intera>])

```
10 x=0.123456789
20 FOR n=9 TO 0 STEP-1
30 PRINT n,ROUND (x,n)
40 NEXT
50 x=123456789
60 FOR r=0 TO -9 STEP -1
70 PRINT r, ROUND(x,r)
80 NEXT
```

FUNZIONE: Arrotonda l'<espressione numerica> ad un numero con posizioni decimali o potenze di dieci specificati nel parametro <espressione intera>. Se <espressione intera> è minore di zero, l'<espressione numerica> viene arrotondata per ottenere un intero assoluto con un numero di zeri uguale al valore specificato prima del punto decimale.

Parole chiave associate: ABS, CINT, FIX, INT.

RUN

RUN <espressione di stringa>

RUN "welcome"

COMANDO: Carica un programma dalla cassetta e ne inizia l'esecuzione. Se l'espressione stringa è vuota (""), BASIC cercherà di caricare il primo file incontrato sul nastro. Se il primo carattere del nome del file è !, i messaggi di gestione della cassetta non verranno visualizzati.
NB: BASIC esegue un vero e proprio NEW ogni volta che viene caricato un programma dalla memoria.

Parole chiave associate: LOAD.

RUN

RUN [<numero linea>]

RUN 100

COMANDO: Inizia l'esecuzione del corrente programma BASIC, dal parametro <numero linea> specificato o dall'inizio del programma se il parametro viene omesso. RUN assegna a tutte le varabili del programma corrente il valore zero o nullo. I parametri DEFINT, DEFREAL e DEFSTR vengono reinizializzati. Tutti i file aperti su cassetta vengono abbandonati e tutto l'output bufferizzato viene perso.

Parole chiave associate: LOAD

SAVE

SAVE <nomefile>[,<tipo file>][,<parametri binari>]

```
SAVE "prog",P
SAVE "BINARY", B, 10000, 16000,10003
```

COMANDO: Salva il programma attualmente in memoria con il nome <nomefile>. I parametri Binari sono: <indirizzo iniziale>,<lunghezza file>[,<punto di ingresso>]

,A salva un programma in ASCII
,P protegge il file
,B salva un'area di memoria (ad esempio lo schermo) come un file binario

Parole chiave associate: CHAIN, CHAIN MERGE, LOAD, MERGE, RUN <nome file>.

SGN

SGN(<espressione numerica>)

```
10 INPUT "Qual'è l'ammontare attuale del tuo conto in banca";lire
20 IF SGN(lire)<1 GOTO 30 ELSE 40
30 PRINT "Ahi Ahi!":END
40 PRINT "Allora hai più soldi di me"
```

FUNZIONE: Determina il SeGNo dell'<espressione numerica>. SGN restituisce -1 se l'<espressione numerica> è minore di zero, 0 se l'<espressione numerica> è uguale a zero e 1 se l'<espressione numerica> è maggiore di zero.

Parole chiave associate: ABS.

SIN

SIN(<espressione numerica>)

```
PRINT SIN(PI/2)
 1
```

FUNZIONE: Calcola il seno dell'<espressione numerica> utilizzando la misura in radianti a meno che sia stato precedentemente specificato in un comando DEG.

Parole chiave associate: ATN, COS, DEG, RAD, TAN.

SOUND

SOUND <stato canale>,<periodo di tono>[,<durata>[,<volume>[,<inviluppo volume>[,<inviluppo tono>[,<periodo rumore>]]]]]

```
SOUND 1,200,1000,7,0,0,1
```

COMANDO: Le possibilità del comando SOUND costituiscono una delle estensioni più complicate del BASIC e sono introdotte nel Capitolo 6.

Parole chiave associate: ENT, ENV

SPACE$

SPACE$(<espressione intera>)

```
SPACE$(190)
```

FUNZIONE: Crea una stringa di spazi della lunghezza data.

Parole chiave associate: PRINT, TAB.

SPEEK INK

SPEED INK <espressione intera 1>,<espressione intera 2>

```
5 INK 0,9,12:INK 1,0,26
10 BORDER 12,9
20 SPEED INK 50,20
```

COMANDO: Determina la frequenza con cui si devono alternare due colori specificati in un comando INK o BORDER. <espressione intera 1> specifica il tempo, in unità di 0.02 secondi (cinquantesimi di secondo) per il primo colore da usare; <espressione intera 2> specifica il tempo per il secondo colore. E' necessario utilizzare unpo' di buon senso per evitare effetti ipnotici quando si scelgono i colori e le frequenze di ripetizione!

Parole chiave associate: BORDER, INK.

SPEED KEY

SPEED KEY <ritardo iniziale>,<periodo ripetizione>

SPEED KEY 20,3

COMANDO: Assegna la frequenza di autoripetizione alla tastiera. Il parametro <ritardo iniziale> specifica il tempo, in unità di 0.02 secondi (cinquantesimi di secondo) prima dell'inizio dell'autoripetizione. Il parametro <periodo ripetizione> assegna l'intervallo tra ciascuna autoripetizione di un tasto. I parametri possono assumere valori tra 1 e 255. Quando si desidera utilizzare piccoli valori di <ritardo inizio>, è possibile che sorgano dei problemi con le routine anti-rimbalzamento della tastiera. Questo comando non modifica la velocità alla quale viene letta la tastiera. Non tutti i tasti possiedono autoripetizione; il comando KEY DEF permetterà di modificare gli attributi di un determinato tasto.

Parole chiave associate: KEY DEF.

SPEED WRITE

SPEED WRITE <espressione intera>

SPEED WRITE 1

COMANDO: Determina la velocità di memorizzazione dei dati su cassetta. Si può scrivere su cassetta alla velocità di 2000 baud (bit per secondo) se l'<espressione intera> è 1, oppure alla velocità per difetto di 1000 baud se l'<espressione intera> è 0. Quando si carica in memoria un file su nastro, il computer seleziona automaticamente la corretta velocità di trasferimento.

Per l'affidabilità di dati importanti, è consigliabile l'uso di SPEED WRITE 0 (valore per difetto). Per ulteriori informazioni si veda il Capitolo 2.

Parole chiave associate: SAVE.

SQ

SQ(<canale>)

```
10 MODE 1
20 FOR n=20 TO 0 STEP -1
30 PRINT n;
40 SOUND 1,10+n,100,7
50 WHILE SQ(1)>127:WEND
60 NEXT
```

FUNZIONE: Viene usata per controllare il numero di posti liberi nella coda di un determinato canale dove il canale A è 1, B è 2 e C è 3. La funzione determina se il canale è attivo e se non lo è, perchè si trova in attesa. Il risultato deve essere letto bit a bit:

Bit 0, 1 e 2	: numero di elementi liberi nella coda
Bit 3, 4 e 5	: stato di rendezvous all'inizio della coda
Bit 6	: l'inizio della coda è bloccato
Bit 7	: il canale è attivo

Parole chiave associate: ON SQ GOSUB, SOUND.

SQR

SQR(<espressione numerica>)

```
PRINT SQR(9)
 3
```

FUNZIONE: Restituisce la radice quadrata dell'<espressione numerica> specificata.

Parole chiave associate: PRINT

STOP

STOP

```
20 IF n<0 THEN STOP
```

COMANDO: Blocca l'esecuzione di un programma, ma lascia BASIC in uno stato in cui il programma può essere riattivato mediante il comando CONT. STOP può essere utilizzato per interrompere il programma in punti particolari durante un'operazione di debugging.

Parole chiave associate: CONT, END.

STR$

STR$(<espressione numerica>)

```
PRINT STR$(&766)
 1894

PRINT STR$(&X1010100)
 84
```

FUNZIONE: Converte l'<espressione numerica> in una rappresentazione decimale di stringa.

Parole chiave associate: BIN$, DEC$, HEX$, VAL.

STRING$

STRING$(<lunghezza>,<specificatore carattere>)

```
PRINT STRING$(&16,"*")
**********************
```

FUNZIONE: Restituisce una espressione di stringa formata dal carattere specificato ripetuto il numero di volte specificato in <lunghezza>.

Parole chiave associate: SPACE$.

SYMBOL

SYMBOL <numero carattere>,<lista di:<riga>

```
5 MODE 2
10 SYMBOL AFTER 90
20 SYMBOL 93,&80,&40,&20,&10,&8,&4,&2,&1
30 FOR n=1 TO 2000
40 PRINT CHR$(93);
50 NEXT
60 GOTO 60
```

COMANDO: Ridefinisce la forma di un carattere (specificato mediante il comando SYMBOL AFTER) sullo schermo. E' possibile utilizzare un qualunque carattere ASCII o un qualunque carattere del set di caratteri del CPC464 ed i numeri successivi specificano la griglia 8x8 del carattere. Un valore uguale a zero indica di usare il colore dello sfondo mentre un valore uguale a 1 indica di usare il colore dell'inchiostro. Vedere anche le Appendici I e III. L'esempio qui sopra produce una barra rovesciata, utilizzabile premendo il tasto].

Parole chiave associate: SYMBOL AFTER.

SYMBOL AFTER

SYMBOL AFTER <espressione intera>

SYMBOL AFTER 90

COMANDO: Assegna il numero di caratteri definibili dall'utente (compresi tra 0 e 256). Il valore per difetto è 240, e consente di definire 16 caratteri. Se l'<espressione intera> è 32, tutti i caratteri da 32 a 255 diventano ridefinibili. Quando viene eseguito un comando SYMBOL AFTER, tutti i caratteri definiti da utente vengono riportati alle loro condizioni per difetto.

Parole chiave associate: SYMBOL

TAG

TAG[#<canale>]

```
10 MODE 2
11 BORDER 9
14 INK 0,12
15 INK 1,0
20 FOR n=1 TO 100
30 MOVE 200+n,320+n
40 TAG
50 IF n<70 GOTO 60 ELSE 70
60 PRINT "Ciao";:GOTO 80
70 PRINT " Amico";
80 NEXT
90 GOTO 20
```

COMANDO: Invia qualsiasi testo specificato nel <canale> alla posizione del cursore grafico: in questo modo testo e simboli possono essere inseriti all'interno di grafici. Se viene omesso il <canale>, viene assunto per difetto il canale #0. L'angolo superiore sinistro del carattere viene accostato al cursore grafico, mentre i caratteri di controllo non stampabili verranno visualizzati. In particolare il carattere di "nuova linea" verrà visualizzato se l'istruzione PRINT non è chiusa da un punto e virgola.

Parole chiave associate: TAGOFF.

TAGOFF

TAGOFF[#<canale>]

TAGOFF #0

COMANDO: Disabilita TAG (testo e grafica) per il <canale> specificato (canale #0 se non indicato) e ridirige il testo alla posizione precedente del cursore di testo usato prima del comando TAG.

Parole chiave associate: TAG.

TAN

TAN(<espressione numerica>)

PRINT TAN(45)

FUNZIONE: Calcola la TANgente dell'<espressione numerica>, che deve essere compresa tra -200000 e +200000 utilizzando la misura in radianti a meno che sia stato esplicitamente indicato diversamente con un comando DEG.

Parole chiave associate: ATN, COS, DEG, RAD, SIN.

TEST

TEST(<coordinata x>,<coordinata y>)

PRINT TEST(300,300)

FUNZIONE: Restituisce il valore dell'inchiostro alla locazione specificata.

Parole chiave associate: MOVE, MOVER, TESTR, PLOT, PLOTR, DRAW, DRAWR.

TESTR

TESTR(<spostamento x>,<spostamento y>)

TESTR(5,5)

FUNZIONE: Porta il cursore grafico alla posizione specificata dagli spostamenti x e y relativi alla sua posizione corrente e restituisce il valore dell'inchiostro alla nuova locazione.

Parole chiave associate: MOVE, MOVER, TEST, PLOT, PLOTR, DRAW, DRAWR

TIME

TIME

```
10 dato=INT(TIME/300)
20 tic=((TIME/300)-dato)
30 PRINT tic;
40 GOTO 20
```

FUNZIONE: Restituisce il tempo trascorso dall'ultima accensione o dall'ultimo riavvio del computer (ad esclusione dei momenti di lettura o scrittura su cassetta). Le unità di tempo sono pari ad 1/300 di secondo.

TROFF
TRON

TROFF
TRON

TRON

COMANDO: BASIC include la possibilità di tener traccia dell'esecuzione di un programma, stampando ogni numero di linea prima di eseguirlo. Il numero di linea compare tra parentesi quadre [ ]. TRON attiva tale azione, mentre TROFF lo disattiva.

Parole chiave associate: RUN

UNT

UNT(<espressione di indirizzo>)

```
PRINT UNT(65535)
-1
```

FUNZIONE: Converte un intero a 16 bit compreso tra 0 e 65535. Restituisce un intero compreso tra -32768 e +32767.

Parole chiave associate: CINT, FIX, INT, ROUND.

UPPER$

UPPER$(<espressione di stringa>)

```
PRINT UPPER$("amstrad")
AMSTRAD
```

FUNZIONE: Restituisce una nuova espressione di stringa che risulta essere una copia dell'<espressione di stringa> specificata in cui tutti i caratteri alfabetici compresi tra A e Z vengono convertiti in lettere maiuscole.

Parole chiave associate: LOWER$.

VAL

VAL(<espressione di stringa>)

```
10 a$="7 è il mio numero fortunato"
20 PRINT VAL(a$)
```

FUNZIONE: Restituisce il VALore numerico del primo carattere (o dei primi caratteri) nella <espressione di stringa> specificata.

Parole chiave associate: STR$.

VPOS

VPOS(#<canale>)

PRINT VPOS(#0)

FUNZIONE: Restituisce la posizione verticale del cursore di testo del canale.

Parole chiave associate: POS

WAIT

WAIT <numero porta>,<maschera>[,<inversione>]

```
WAIT &FF34,20,25
```

COMANDO: Attende finchè il <numero porta> di I/O specificato non restituisce un particolare valore compreso tra 0 e 255. BASIC resta in ciclo durante la lettura della porta di I/O. Il valore letto viene sottoposto ad un'operazione di OR esclusivo con l'<inversione> e quindi ad un'operazione di AND con la <maschera> finchè non si ottiene un risultato diverso da zero. BASIC attenderà indefinitamente fino al verificarsi della condizione richiesta. Immettendo l'esempio proposto, occorrerà reinizializzare il computer per poterlo riutilizzare.

Parole chiave associate: INP, OUT.

WEND

WEND

```
10 MODE 1:REM Orologio in BASIC
20 INPUT "Immettere ora, minuto e secondo (o,m,s)";ora, min, sec
30 CLS:dato=INT(TIME/300)
40 WHILE ora<13
50 WHILE min<60
60 WHILE tic<60
70 tic=(INT(TIME/300)-dato)+sec
80 LOCATE 70,4
90 PRINT#0, USING "## ";ora, min, tic
100 WEND
110 tic=0
120 sec=0
130 min=min+1
140 GOTO 30
150 WEND
160 min=0
170 ora=ora+1
180 WEND
190 ora=1
200 GOTO 40
```

COMANDO: Il ciclo WHILE/WEND esegue ripetutamente una parte del programma fino al verificarsi di una data situazione. L'esempio illustra l'eleganza dei programmi costruiti in questo modo. WEND segnala la fine del ciclo WHILE.
Parole chiave associate: WHILE.

WHILE

WHILE <espressione logica>

WHILE DAY<0

COMANDO: Esegue ripetutamente il corpo di un programma mentre una data condizione risulta vera. Il comando WHILE definisce l'inizio del ciclo e specifica la condizione nell'<espressione logica>. Il comando WEND termina il ciclo.

Parole chiave associate: WEND.

WIDTH

WIDTH <espressione intera>

```
WIDTH 86
```

COMANDO: Indica a BASIC quanti caratteri per linea devono essere stampati quando si collega una stampante. BASIC invierà quindi i caratteri di ritorno carrello o line feed quando necessario.

Parole chiave associate: POS.

WINDOW

WINDOW[#<canale>,]<sinistra>,<destra>,<alto>,<basso>

```
10 MODE 1
20 BORDER 6
30 WINDOW 10,30,7,18
40 PAPER 2:PEN 3
50 CLS
60 PRINT CHR$(143);CHR$(242);"Posizione"
70 PRINT "1,1 nella finestra di testo"
80 GOTO 80
```

COMANDO: Specifica una finestra per un dato canale.

Parole chiave associate: ORIGIN.

WINDOW SWAP

WINDOW SWAP <canale1>,<canale2>

```
WINDOW SWAP 0,2
```

COMANDO: Scambia la finestra di testo specificata in <canale1> con quella specificata in <canale2>. Ad esempio i messaggi inviati da BASIC al canale #0 possono essere inviati ad un altro canale.

Parole chiave associate: WINDOW, PAPER, PEN, TAG.

WRITE

WRITE [#<canale>,][<lista scrittura>]

```
WRITE #2,"Ciao",4,5
"Ciao",4,5
```

COMANDO: Scrive i valori degli elementi della <lista scrittura> sul canale specificato separati da virgole; le stringhe sono inserite tra doppi apici.
E' usato soprattutto per scrivere dati su file su cassetta.

XPOS

XPOS

PRINT XPOS

FUNZIONE: Restituisce la corrente POSizione orizzontale (X) del cursore grafico.

Parole chiave associate: MOVE, MOVER, ORIGIN, YPOS.

YPOS

YPOS

PRINT YPOS

FUNZIONE: Restituisce la corrente POSizione verticale (Y) del cursore grafico.

Parole chiave associate: MOVE, MOVER, ORIGIN, XPOS.

ZONE

ZONE <espressione intera>

```
10 PRINT 1,2,3
20 ZONE 19
30 PRINT 4,5,6
```

COMANDO: Modifica le dimensioni della zona di stampa (specificata nelle istruzioni PRINT mediante una virgola tra gli elementi). Il valore per difetto della zona di stampa è di 13 colonne, ma può essere modificata come indicato nella <espressione intera>, che deve essere compresa tra 1 e 255. Viene reinizializzato da NEW, LOAD, CHAIN e RUN "<nome file>".

Parole chiave associate: PRINT, WIDTH.

PRINT

PRINT[#<canale>,][<lista di stampa>][<parte USING>][<separatore>]

<lista di stampa> è <oggetto>[<separatore><lista di stampa>]
<oggetto> è una <espressione>
o SPC(<espressione intera>)
TAB(<espressione intera>)

Scrive dati su un file su cassetta.

```
10 OPENOUT "DATI"
20 PRINT #9, "Ciao"
30 CLOSEOUT
```

Invia dati ad una stampante

```
10 valore=23000*PI
20 PRINT #8, USING "######.##";valore
30 PRINT #0, valore
run
 72256.6311
Ready
```

[Sulla stampante invece...]

```
 72256.63
```

COMANDO: Stampa dati sul canale specificato (sul canale #0 se non viene specificato un <canale>).

Se non si specifica alcun formato, BASIC stampa in un "formato libero" in cui se un oggetto è seguito da una virgola, l'oggetto successivo verrà stampato nella zona successiva (per difetto 13). Un punto e virgola separa semplicemente le espressioni.

SPC(<espressione intera>) stampa il numero di spazi indicati assumendo 0 spazi se l'<espressione intera> è minore di 1. Non è necessario concludere SPC con una virgola o un punto e virgola verrà automaticamente assunto un punto e virgola.

TAB(<espressione intera>) stampa il numero di spazi che occorrono per spostarsi in una determinata posizione di stampa, se l'espressione è minore di 1 viene assunto 1.

Se la posizione corrente è maggiore della posizione richiesta, vengono inseriti gli spazi necessari per raggiungere la posizione richiesta sulla linea - se è minore, viene inviato un carattere di ritorno carrello, seguito dagli spazi necessari per raggiungere la posizione desiderata. Non occorre terminare TAB con una virgola o un punto e virgola.

(continua...)

PRINT *(continua ...)*

PRINT USING "<specificatori del formato>"

All'interno del numero:

Caratt.	Cifre Poss.	Caratteri Campo	Definizione	Esempio
#	1	1	Campo numerico	####
.	0	1	Punto decimale	#.#
+	0	1	Stampa del segno iniziale o finale	+##
			i numeri positivi hanno un +	###+
-	0	1	Stampa del segno: - se negativo altrimenti nulla	##.##-
**	2	2	Asterischi in testa	**##.##
$$	1	2	Simbolo di dollaro prima del numero	$$##.##
**$	2	3	Asterischi e simbolo di dollaro	**$#.##
,	1	1	Virgola ogni tre cifre	###,###,#
↑ ↑ ↑ ↑	0	4	Forma esponenziale. I numeri sono allineati in modo che la prima cifra non sia zero	#,## ↑ ↑ ↑ ↑

STRINGHE

Caratt.	Definizione	Esempio
!	Solo primo carattere	!
\<spazi>\	Alcuni spazi più un altro spazio	\\
&	Campo a lunghezza variabile	&

9 Ulteriori informazioni di programmazione

Gli argomenti affrontati in questo capitolo sono:

* Posizionamente del testo
* Caratteri di controllo
* Sistema operativo
* Meccanismi di interruzione

9.1 Posizione del cursore ed estensione dei codici di controllo

In molti programmi applicativi, il cursore del testo può essere posizionato all'esterno della finestra attuale. Varie operazioni costringeranno il cursore ad una posizione lecita:

- Scrivere il carattere
- Disegnare il cursore
- Eseguire i codici di controllo segnati da un asterisco nella lista che segue.

La procedura che porta il cursore in una posizione lecita è la seguente:

a. Se il cursore è a destra del margine destro, viene portato alla colonna più a sinistra della linea seguente.

b. Se il cursore è a sinistra del margine sinistro, viene portato alla colonna più a destra della linea precedente.

c. Se il cursore è sopra il margine superiore, la finestra viene fatta scorrere di una linea ed il cursore viene portato sulla riga più in alto nella finestra.

d. Se il cursore è sopra il margine inferiore, la finestra viene fatta scorrere di una linea ed il cursore viene portato sulla riga più in basso nella finestra.

I controlli e le operazioni vengono effettuate esattamente nell'ordine riportato. Le posizioni illegali del cursore possono essere uguali a zero o negative (che si trovano a sinistra o al di sopra dello schermo).

I caratteri tra 0 e 31 inviati allo schermo non producono un carattere ma vengono interpretati come codici di controllo (e come tali non devono essere impiegati a sproposito). Alcuni codici modificano il significato dei caratteri seguenti che costituiscono i parametri del codice.

I codici indicati da un asterisco (*) costringono il cursore ad una posizione lecita nella finestra corrente prima di essere eseguiti ma possono lasciare il cursore in una posizione non lecita. I codici ed il loro significato sono descritti prima dal loro valore in esadecimale (&XX) e poi dal corrispondente decimale.

Caratteri di controllo BASIC

Valore		Nome	Parametro	Significato
&00 0		NUL		Nessun effetto. Ignorato
&01 1		SOH	da 0 a 255	Stampa il simbolo indicato dal parametro. Permette di visualizzare i simboli tra 0 e 31.
&02 2		STX		Elimina il cursore del testo.
&03 3		ETX		Riattiva il cursore del testo. Si noti che BASIC usa una disabilitazione da tastiera la quale viene rilasciata solo quando la tastiera attende un input.
&04 4		EOT	da 0 a 2	Fissa la modalità di schermo. Parametro MOD 4. Equivalente al comando MODE.
&05 5		ENQ	da 0 a 255	Invia il carattere indicato dal parametro allo schermo grafico.
&06 6		ACK		Abilita lo schermo di testo (Vedere &15 NAK)
&07 7		BEL		Emette un Bip. Svuota le code relative ai suoni.
&08 8	*	BS		Sposta il cursore indietro di un carattere.
&09 9	*	TAB		Sposta il cursore avanti di un carattere.
&0A 10	*	LF		Sposta il cursore verso il basso di una linea.
&0B 11	*	VT		Sposta il cursore verso l'alto di una linea.
&0C 12		FF		Cancella la finestra di testo e porta il cursore nell'angolo in alto a sinistra. Equivale al comando CLS.
&0D 13	*	CR		Porta il cursore al primo carattere a sinistra della stessa linea.
&0E 14		SO	da 0 a 15	Fissa il colore della carta. Parametro MOD 16. Equivale al comando PAPER.
&0F 15		SI	da 0 a 15	Fissa il colore della penna. Parametro MOD 16. Equivale al comando PEN.
&10 16	*	DLE		Cancella il carattere corrente. Riempie la griglia che costituisce il carattere con il colore della carta.

Valore		Nome	Parametro	Significato
&11 17	*	DC1		Cancella la linea dal margine sinistro della finestra fino alla posizione (inclusa) del cursore. Riempie le posizioni interessate con il colore della carta.
&12 18	*	DC2		Cancella la linea dalla posizione (inclusa) del cursore al margine destro della finestra. Riempie le posizioni interessate con il colore della carta.
&13 19	*	DC3		Cancella il contenuto della finestra fino alla posizione del cursore. Riempie le posizioni interessate con il colore della carta.
&14 20	*	DC4		Cancella il contenuto della finestra dalla posizione del cursore in poi. Riempie le posizioni interessate con il colore della carta.
&15 21		NAK		Disattiva lo schermo di testo. Lo schermo non reagirà ad alcun comando ad esso inviato fino alla ricezione di un carattere ACK (&06 6).
&16 22		SYN	da 0 a 1	Parametro MOD 2. Opzione Trasparenza. 0 disabilita 1 abilita.
&17 23		ETB	da 0 a 3	Parametro MOD 4. 0 fissa il normale modo grafico 1 fissa il modo grafico XOR 2 fissa il modo grafico AND 3 fissa il modo grafico OR
&18 24		CAN		Scambia i colori di carta e penna.
&19 25		EM	da 0 a 255 da 0 a 255 da 0 a 255 da 0 a 255 da 0 a 255 da 0 a 255 da 0 a 255 da 0 a 255 da 0 a 255	Riempie la matrice di un carattere definibile dall'utente. Equivale al comando SYMBOL. Prende 9 parametri. Il primo indica il carattere da riempire. Gli altri 8 specificano la matrice. Il bit più significativo del primo byte cor-risponde al pixel in alto a sinistra della cella del carattere, il bit meno significativo dell'ultimo byte corrisponde al pixel in basso a destra della cella del carattere
&1A 26		SUB	da 1 a 80 da 1 a 80 da 1 a 25 da 1 a 25	Fissa la finestra. Equivale al comando WINDOW. I primi due parametri specificano i margini sinistro e destro della finestra (il valore più piccolo viene considerato il margine sinistro, il più grande il margine destro). Gli altri due parametri specificano i margini superiore ed inferiore della finestra (il valore più piccolo viene considerato il margine superiore, il più grande il margine inferiore

Valore	Nome	Parametro	Significato
&1B 27	ESC		Nessun effetto. Ignorato.
&1C 28	FS	da 0 a 15 da 0 a 31 da 0 a 31	Fissa l'inchiostro ad una coppia di colori. Equivale al comando INK. Il primo parametro (MOD 16) specifica l'inchiostro, gli altri due (MOD 32) i colori richiesti. i valori compresi tra 27 e 31 corrispondono a colori non definiti.
&1D 29	GS	0..31 0..31	Fissa il bordo a due colori. E' equivalente al comando BORDER. I due parametri specificano (MOD 32) i due Colori.
&1E 30	RS		Porta il cursore nella posizione in alto a sinistra della finestra.
&1F 31	US	da 1 a 80 da 1 a 25	Porta il cursore nella posizione indicata nella finestra corrente. Equivale al comando LOCATE. Il primo parametro costituisce la colonna su cui portarsi, il secondo corrisponde alla linea.

9.2 Sistema operativo

La gestione del 464 è dotata di un sofisticato sistema operativo in tempo reale. Il sistema operativo "dirige il traffico" tra input ed output.

Innanzitutto fornisce l'interfaccia tra l'hardware e l'interprete BASIC (ad esempio nel caso del colore lampeggiante, dove il BASIC passa semplicemente dei parametri ed il sistema operativo esegue l'ordine; in tal modo una parte decide cosa deve essere fatto e l'altra come ciò deve essere fatto.

Il sistema operativo della macchina è conosciuto generalmente come il "firmware" ed è composto dalle routine in codice macchina chiamate dai comandi ad alto livello del BASIC.

Se si è tentati di fare delle POKE negli indirizzi della memoria della macchina o effettuare delle CALL alle subroutine, è conveniente salvare prima il programma contenuto in memoria; dopo tali operazioni è possibile perderlo. Le grandi potenzialità del firmware contenente il sistema operativo del 464 sono descritte nella guida utenti avanzata; la loro spiegazione va oltre lo scopo del manuale introduttivo.

9.3 : Interrupts (Interruzioni)

Il CPC 464 fa un uso estensivo degli interrupt dello Z80 e fornisce un sistema operativo che include molte possibilità di multi-tasking, esemplificate dalle strutture AFTER e EVERY descritte precedentemente in questo manuale. La precedenza degli eventi :

Break (**[ESC][ESC]**)
Timer 3
Timer 2 (e le tre code del suono)
Timer 1
Timer 0

Le interruzioni dovrebbero essere incluse solo dopo aver considerato le conseguenze dei possibili stati intermedi delle variabili al momento dell'interruzione. La stessa sub-routine di interruzione dovrebbe evitare interazioni indesiderate con lo stato delle variabili del programma principale.

Le code del suono hanno interruzioni indipendenti di uguale priorità. Quando inizia una interruzione del suono non può essere interrotta da nessun'altra interruzione del suono. Ciò permette di far condividere le variabili alle routine di interruzione del suono senza che sorga alcun problema.

Quando l'interruzione di una coda del suono è abilitata agirà immediatamente se la coda del suono di quel canale non è piena, altrimenti agirà quando termina il suono attualmente prodotto e si crea uno spazio nella coda. L'interruzione disabiliterà l'evento, in modo che la sub-routine possa riabilitarsi se sono richieste altre interruzioni.

Cercando di produrre un suono o controllando lo stato della coda si disabilita una interruzione del suono.

La priorit della sequenza **[ESC][ESC]** su tutte le altre interruzioni assicura che le operazioni BASIC non possano essere alterate senza perdita di programma: non fornendo nessuna azione ausiliare si assicura l'integrità del programma mediante diverse tecniche di protezione.

9.3 Assembler AMSTRAD

Per scrivere programmi in codice macchina necessario usare un Assembler. L'assembler AMSTRAD comprende un assemblatore Z80, con editor, disassembler e monitor.

10 Interrupt (Interruzioni)

Argomenti trattati in questo capitolo:

* AFTER
* EVERY
* REMAIN
* L'orologio principale

Forse si sarà già notato che una delle innovazioni più importanti nel software del CPC464 è costituito dalla possibilità di gestire gli interrupt da BASIC; questo significa che il BASIC riesce ad eseguire simultaneamente ed indipendentemente un certo numero di operazioni all'interno di un programma. Tale possibilità è nota con il nome di multi-tasking ed è implementata dai due comandi AFTER ed EVERY.

Questa possibilità è chiaramente dimostrata dal modo con cui è possibile gestire il suono mediante code e rendez-vous.

Ogni aspetto della sincronizzazione si riferisce all'orologio principale che è un sistema di sincronizzazione al quarzo contenuto nel computer principale. Esso occupa del tempo e della sincronizzazione degli eventi che avvengono nel computer - operazioni come la scansione dello schermo ed il clock del processore. Quando una funzione dell'hardware è connessa al tempo, può essere seguita utilizzando l'orologio principale al quarzo.

L'implementazione software è costituita dai comandi AFTER e EVERY che, secondo l'approccio rivolto verso l'utente del BASIC AMSTRAD, fanno esattamente ciò che dicono: ad esempio AFTER (DOPO) il momento indicato nella linea di comando, il programma deve eseguire una determinata sub-routine le operazioni indicate.

10.1 AFTER

Il CPC 464 utilizza un orologio in tempo reale. Il comando AFTER permette di eseguire da BASIC delle sub-routine in un dato momento del futuro. Sono disponibili quattro timer di ritardo ognuno dei quali può avere associata una sub-routine.

AFTER <espressione intera>[,<espressione intera>] GOSUB <numero di linea>

La prima <espressione intera> specifica tra quanto tempo deve essere chiamata la sub-routine. Questo tempo è misurato in cinquantesimi di secondo.

La seconda <espressione intera> specifica quale timer di ritardo usare. L'espressione deve fornire un valore tra 0 e 3. Se non si specifica questa espressione, viene assunto 0.

Quando il tempo specificato è trascorso, la sub-routine viene automaticamente chiamata, proprio come se fosse stato eseguito un GOSUB nella posizione corrente del programma. Quando la sub-routine termina, mediante un normale comando RETURN, il programma continua la propria esecuzione da dove era stato interrotto.

I timer hanno diverse priorità di interruzione. Il timer 3 ha la proprietà più alta ed il timer 0 quella più bassa.

Il comando AFTER può essere eseguito in qualunque momento, reinizializzando la routine ed il tempo di ritardo associato ad un dato timer. I timer di ritardo sono gli stessi usati nel comando EVERY, e quindi un AFTER sostituisce un precedente EVERY che utilizza un determinato timer e viceversa.

```
10 MODE 1:X=0
20 AFTER 45 GOSUB 100
30 AFTER 100,1 GOSUB 200
40 PRINT "AMSOFT"
50 WHILE X<100
60 LOCATE #1,30,1:PRINT #1,X:X=X+1
70 WEND
80 END
100 PRINT "peripherals"
110 RETURN
200 PRINT "and software"
210 RETURN
```

Si noti l'uso dei due canali (finestre) che permette di stampare il programma principale (linee 50-80) usando un posizionamento del cursore indipendente da quello usato dalle sub-routine degli interrupt.

AFTER (DOPO) che è trascorso il tempo specificato nella linea di comando, il programma esegue la sub-routine specificata.

10.2 EVERY

Il comando EVERY permette di chiamare delle sub-routine da BASIC ad intervalli regolari. Sono disponibili quattro timer di ritardo ognuno dei quali può avere associata una sub-routine.

EVERY <espressione intera>[,<espressione intera>] GOSUB <numero di linea>

La prima <espressione intera> specifica tra quanto tempo deve essere chiamata la sub-routine. Questo tempo è misurato in cinquantesimi di secondo.

La seconda <espressione intera> specifica quale timer di ritardo usare. L'espressione deve fornire un valore tra 0 e 3. Se non si specifica questa espressione, viene assunto 0.

Quando il tempo specificato è trascorso, la sub-routine viene automaticamente chiamata, proprio come se fosse stato eseguito un GOSUB nella posizione corrente del programma. Quando la sub-routine termina, mediante un normale comando RETURN, il programma continua la propria esecuzione da dove era stato interrotto.

I timer hanno diverse priorità di interruzione. Il timer 3 ha la proprietà più alta ed il timer 0 quella più bassa. Quando un timer termina, viene iniziato un nuovo conto per la prossima chiamata della sub-routine.

Il comando EVERY può essere eseguito in qualunque momento, reinizializzando la routine ed il tempo di ritardo associato ad un dato timer. I timer di ritardo sono gli stessi usati nel comando AFTER, e quindi un EVERY sostituisce un precedente AFTER che utilizza un determinato timer e viceversa.

```
10 MODE 1:X=0
20 P100=0:EVERY 10 GOSUB 100
30 P200=0:EVERY 12,1 GOSUB 200
40 PRINT "AMSOFT"
50 WHILE X<200
60 LOCATE #1,30,1:PRINT #1,X:X=X+1
70 WEND
80 LOCATE 1,20:END
100 DI:PEN P100:LOCATE 1,2:PRINT "peripherals":EI
105 IF P100=0 THEN P100=1 ELSE P100=0
110 RETURN
200 PEN P200:LOCATE 1,3:PRINT "and software"
205 IF P200=2 THEN P200=3 ELSE P200=2
210 RETURN
```

Si noti l'uso dei comandi DI che disabilita ed EI che`abilita i timer e gli interrupt per il suono mentre i comandi all'interno di essi vengono eseguiti. Questo ha l'effetto di ritardare l'interrupt di proprietà più alta del timer 1 da tutto ciò che potrebbe accadere durante la gestione dell'interrupt da parte del timer 0 (linee 100-110). Perciò i valori fissati dai comandi comandi PEN e LOCATE non vengono modificati prima dell'esecuzione del comando PRINT.

10.3 REMAIN

Questa funzione restituisce il tempo mancante per uno dei quattro timer. Disabilita il timer e restituisce zero se il timer è già disabilitato. Viene usato nella forma:

REMAIN(<espressione intera>)

Appendice I

L'Arte del possibile

I principianti del calcolo iniziano frequentemente instaurando alcuni punti di riferimento che aiuteranno a fornire loro una idea della capacità e delle limitazioni del computer. Questo paragrafo ne spiega i concetti in una ampia guida.

Anche se si è acquistato il CPC464 solo per i giochi disponibili si potrebbero voler apprendere ulteriori aspetti su ciò che viene definito 'hardware'.

L'hardware è tutto ciò che si può prelevare e trasportare: la tastiera del computer, il monitor, le connessioni, ecc. In effetti è tutto ciò che non viene chiamato software - programmi, manuali, cassette.

Alcune caratteristiche di comportamento del computer vengono rese disponibili grazie all'hardware; si pensi ad esempio al colore di un televisore o del monitor. E' qundi compito del software utilizzare l'hardware in modo da produrre caratteri e disegni sullo schermo.

E' l'hardware che conduce il fascio di elettroni sulla superficie luminescente dello schermo o del televisore; il software aggiunge ordine ed intelligenza indicando all'hardware quello che deve fare o come deve farlo. Aggiunge sincronizzazione, controllo e sequenza al fine ad esempio di produrre una astronave in partenza o qualcosa di più terrestre come i caratteri che appaiono sullo schermo quando vengono premuti dei tasti.

Cosa rende un computer migliore di un altro?

L'hardware senza il software non ha valore. Il software senza l'hardware è anch'esso senza valore: il vero valore di un computer si scopre quando entrambi interagiscono.

Le caratteristiche principali di un personal computer sono:

1. Risoluzione dello schermo - il più piccolo oggetto discernibile sullo schermo.

Questa è data una combinazione di fattori, incluso il numero di colori diversi disponibili per il programmatore, il numero di aree che possono essere usate per la visualizzazione (i pixel) e il numero di caratteri di testo che possono essere visualizzati su uno schermo.

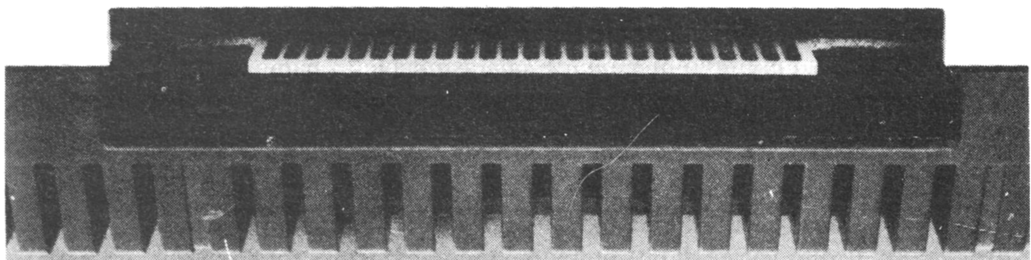
Si potrà notare che il CPC464 è ben dotato rispetto ad altri computer di simile prezzo.

2. Interprete BASIC

Tutti i personal computer, praticamente, includono un interprete BASIC che permette all'utente di creare programmi ed usare le caratteristiche hardware. Il linguaggio di programmazione BASIC del CPC è esso stesso un programma complicato che è stato sviluppato da molte persone negli Stati Uniti. Il 'Beginners All-purpose Symbolic Instruction Code' è il linguaggio maggiormente usato nel mondo, e come tutti i linguaggi ha diversi 'dialetti'.

La versione del CPC464 è una delle più compatibili e potrà eseguire molti dei programmi scritti per il sistema operativo CP/M. E' una velocissima implementazione del BASIC; esegue in altre parole calcoli in maniera velocissima: un computer può impiegare 0.005 secondi per moltiplicare 3 per 5 mentre un altro può impiegarne 0.075; un programma che disegna grafici può impiegare 0.05 secondi per ripetere cento volte un calcolo ripetitivo, un altro computer potrebbe impiegarne 0.075: pochi secondi in più sono una considerevole differenza di prestazioni.

Si potrà spesso sentir parlare di 'codice macchina'. Il codice macchina è una forma di istruzione in codice che viene inviata al processore. Questa richiede meno tempo per essere elaborata aumentando così da 5 a 15 volte la velocità rispetto ad un normale interprete BASIC. Ma ci si impiega dalle 5 alle 50 volte in più per scrivere un programma in codice macchina che non scrivendolo in BASIC.
Il BASIC AMSTRAD è tra i più veloci e ben dotati: incorpora molte caratteristiche di aiuto per coloro che non sono esperti di programmazione ad 'alto livello' e che possono in tal modo ottenere effetti visivi e sonori sorprendenti.



3 Espandibilità

Molti computer hanno bisogno di ulteriori strumenti hardware: stampanti, joystick, dischi. Paradossalmente la maggior parte dei computer domestici più venduti ha bisogno di 'interfacce di espansione' prima di poter installare una stampante o un joystick.

L'acquirente non sempre pensa all'immediato futuro, infatti una macchina che include una stampante parallela (Centronics compatibile) e joystick sarebbe in prospettiva una scelta più economica.

Tra le caratteristiche del CPC464 va inclusa la porta Centronics, la possibilità di aggiungere due joystick, una uscita del suono stereo, ecc, oltre ad un bus di espansione che può essere usato per collegare i drive, l'espansione ROM, le interfacce seriali (RS232).

La ROM (Read Only Memory - Memoria a sola lettura) è un circuito integrato che contiene un programma. Il BASIC fornito con il computer memorizzato in tale ROM ed è possibile aggiungere ulteriori programmi su ROM o sostituire completamente la sua funzione.

I giochi usano una 'cartuccia' software. Una 'cartuccia' è semplicemente una ROM (sotto forma di circuito integrato che memorizza un programma) in una custodia di plastica che può essere collegata e disconnessa in modo molto semplice. Quindi la ROM fornisce le stesse informazioni delle cassette. Comunque le informazioni contenute in ROM vengono caricate in modo velocissimo rispetto al tempo che si impiega a caricarle da cassetta.

Una ROM non può essere usata per memorizzare informazioni che poi debbano essere trasportate in un altro computer così come si fa per le cassette.

Espansione significa assicurare al computer la maggior parte degli sviluppi futuri in software e periferiche. Il sistema CPC464 ha una completa ed intera capacità di espansione.

4 Suono

Le caratteristiche sonore di un computer determinano se il computer suona in maniera sorda o produce una accettabile rappresentazione degli strumenti musicali.

Il CPC464 usa un generatore sonoro a tre canali e 8 ottave ; può quindi produrre una accettabile qualità sonora con un controllo dell'inviluppo dell'ampiezza e del tono. Inoltre, il suono è in configurazione stereo dove un canale fornisce l'uscita sinistra e l'altro la destra: il terzo canale risiede tra i due.

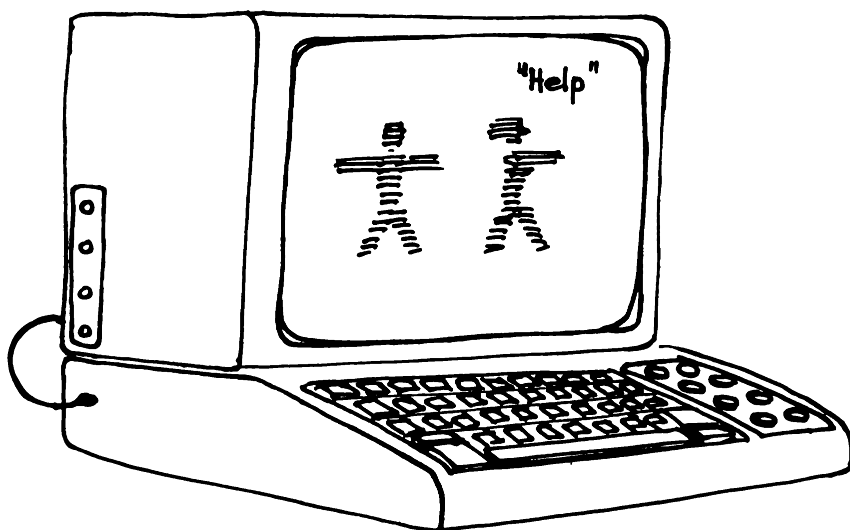
Ciò fornisce una vasta gamma di possibilità di effetti sonori quando ad esempio si scrivono programmi di giochi.

Abbiamo terminato di fornire una idea delle caratteristiche: deciderà l'utente se sono importanti per lui. Speriamo comunque che l'utente le voglia provare ed ottenere il massimo dal computer.

Cosa posso fare?

Con tutte le potenti tecnologie elettroniche gli utenti si chiedono frequentemente se possono visualizzare ad esempio i tipi di figura che appaiono su un televisore.

Perchè un computer non può animare una figura, come ad esempio un uomo che cammina, sullo schermo in modo naturale? Perchè tutti i computer rappresentano i movimenti con figure che sembrano 'robot'?



La risposta è complessa. Una risposta semplice sarebbe quella di dire che lo schermo di un computer non ha nulla a che fare con lo schermo di un televisore. Una televisione opera usando informazioni lineari che descrivono un'area infinita di risoluzioni tra gli estremi 'luce' e 'buio' di tutti i colori di uno spettro. Questo processo, in termini del computer, implica che la 'memoria' di visualizzazione di una figura sul televisore è 20 volte maggiore rispetto ad una analoga sul video del computer.

Questa è solo una parte del problema, poichè l'animazione di una figura richiede una enorme quantità di memoria che deve essere elaborata ad una velocità elevata (50 volte al secondo). Tutto ciò può essere svolto da macchine che costano molti milioni in più di un normale personal computer!

Fino a quando il costo della memoria non crollerà significativamente i piccoli computer dovranno avere a che fare con (relativamente) piccole quantità di memoria per controllare lo schermo: ciò comporterà una bassa risoluzione e movimenti 'goffi'. Sebbene un hardware ben progettato possa fare il meglio non potrà comunque competere con i computer che possono produrre movimenti del tipo dei cartoni animati.

Perchè non si può accendere il computer e scrivere immediatamente e semplicemente una pagina di testo?

Non ci si faccia ingannare dal fatto che la tastiera del computer assomiglia a quella di una macchina per scrivere. Lo schermo non è un foglio elettronico, è una console di comando: questo significa che aiuta a far comunicare l'utente con la memoria della macchina mediante un linguaggio di programmazione.

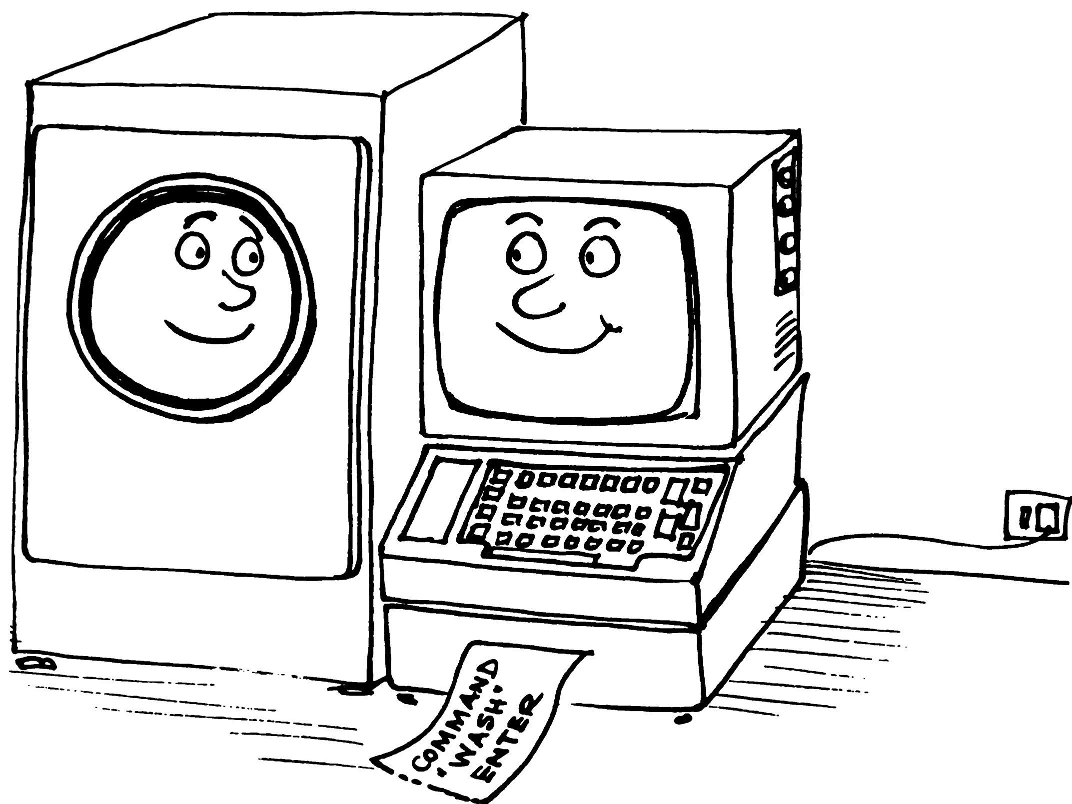
Fino a quando non gli si indichi di fare diversamente, il computer cercherà di eseguire tutto quello che viene immesso alla tastiera come se fosse una istruzione di un programma. Quando si preme il tasto **[ENTER]** il computer cercherà di scoprire se in ci che si inserito vi è qualche cosa che abbia senso per il BASIC in caso contrario commenterà:

Syntax error

Comunque potrebbe accadere che il programma residente in memoria sia in realtà un elaboratore di testi, in tal caso è possibile inserire parole a caso, premere **[ENTER]** e scrivere tutto ciò che si vuole come se lo si facesse su una macchina per scrivere.

Ma per far ciò si deve aver caricato in memoria un programma elaboratore di testi usando il registratore.

Un computer 'sembra' unire insieme diversi oggetti di uso quotidiano come il televisore, la macchina per scrivere, il registratore: occorre ricordarsi comunque che per queste similitudini sono superficiali e che il computer ha una personalità completamente diversa.



GLOSSARIO DEI TERMINI

Alcuni dei termini usati più comunemente in ambiente informatico spiegato agli utenti del CPC464...

A/D

Analogico

Uno stato in cui la modifica tra l'inizio e la fine di un punto accade gradualmente invece che con passi istantanei. I computer sono dispositivi digitali; il mondo naturale è basato su principi analogici quindi, il computer deve eseguire una conversione da analogico a digitale (A/D) prima di poter elaborare i dati di una sorgente analogica.

Accoppiatore acustico

Noto anche come Modem acustico. E' uno strumento elettronico che serve a connettere un telefono al computer ed abilita quest'ultimo a comunicare mediante la normale linea telefonica. In tal modo il computer può comunicare con un servizio di informazione pubblica come Videotel, o con altri utenti per scambiare software ed ottenere dati.

Accumulatore

Una locazione di memoria all'interno del microprocessore, che memorizza temporaneamente i dati mentre vengono elaborati. Sono molto usati nella programmazione in codice macchina; non è necessario per gli utenti di BASIC essere a conoscenza della loro esistenza.

Alfabetizzazione informatica

Un'altra espressione pomposa che significa conoscenza dei computer.

Alfanumerico

Gli attributi che descrivono la differenza tra lettere o numeri e caratteri grafici.

Algebra booleana

E' una istruzione di relazione logica che può assumere solo due forme: vera o falsa. Questi ultimi sono denotati di solito 0 o 1.

Algoritmo

Un nome pomposo per una formula complicata. E' una sequenza di passi aritmetici e logici al fine di eseguire un calcolo.

Alimentatore

Permette di utilizzare la corrente di una normale presa per alimentare il computer.

ALU

Arithmetic Logic Unit (Unità Logico Aritmetica). E' la parte di microprocessore che esegue le operazioni aritmetiche e logiche: non riguarda l'utente direttamente ad eccezione dei programmatori in linguaggio macchina.

Animazione

I cartoni animati sono la forma meglio conosciuta di animazione; l'animazione su un computer è basata sul concetto di movimento di elementi grafici per simulare l'idea del movimento 'reale'.

Architettura

Il modo particolare di un computer di collegare una periferica alla CPU.

Archivio

Un insieme di qualsiasi tipo di dati in un formato indirizzabile dal computer.

Argomento

Una variabile indipendente, ovvero nell'espressione x+y=z, i termini x,y e z sono argomenti.

ASCII

American Standard Code for Information Interchange. Il modo più comunemente usato per rappresentare numeri, lettere ed altri simboli che possono essere inseriti dalla tastiera del computer o richiamati usando altri comandi. I codici del CPC464 sono elencati nell'Appendice III.

Assembler

Un metodo di programmazione in cui le istruzioni in linguaggio macchina vengono richiamate in modo mnemonico (lettere che suggeriscono la funzione che deve essere eseguita dalla corrispondente routine in linguaggio macchina).

Base

L'aspetto numerico più importante per qualunque matematico. La base di ogni rappresentazione numerica. Il sistema binario ha base 2; il sistema decimale ha base 10 e quello esadecimale ha base 16: per ulteriori spiegazione si guardi l'Appendice II.

BASIC
Beginner's All-purpose Symbolic Instruction Code. Un linguaggio di programmazione interpretativo usato in quasi tutti i computer domestici. BASIC è stato progettato in modo specifico per essere semplice da apprendere e facile da usare poichè permette ai programmi di essere 'uniti' tra loro e testati (eseguiti) a qualsiasi stadio del loro sviluppo al contrario dei compilatori nei quali il programma, prima di essere eseguito, deve essere stato completato.

Baud
Un bit per secondo: è l'unità di misura della velocità alla quale un segnale digitale viene trasmesso in sistemi di comunicazione seriali.

BCD
Binary Coded Decimal. Un sistema di codifica per numeri decimali nel quale ogni cifra viene rappresentata come un gruppo di quattro cifre binarie.

Binario
(Si veda Base) Il sistema numerico di base 2 nel quale tutti i numeri sono composti dalle cifre 0 e 1.

Bit
Abbreviazione di BInary digiT (ovvero cifra binaria). Una cifra binaria è composta da 0 e 1 ed è usata nel sistema numerico binario.

Bit meno significativo
In un numero binario (vedere Appendice II) il bit meno significativo (Least Significant Bit o LSB) è il bit all'estrema destra di una espressione.

Bit più significativo
In un numero binario (vedere Appendice II) il bit più significativo (Most Significant Bit o MSB) è il bit all'estrema sinistra di una espressione.

Buffer
Un'area di memoria passeggera contenente informazioni nel corso di un trasferimento di queste da una parte all'altra del sistema, ad esempio da una cassetta all'unità centrale del computer (CPU) e nella RAM. Un buffer regola il modo con cui i dati vengono inviati al dispositivo operante ad una velocità diversa come, ad esempio, un modem o una stampante.

Bug
Un problema che varia da un 'comportamento inaspettato' basato su alcuni aspetti oscuri di un programma (come ad esempio se si premono quattro tasti contemporaneamente si modifica il colore dello schermo) ad una sequenza che rovina irrevocabilmente e completamente un programma o cancella la memoria di dati.

Bus

Un gruppo di connessioni che si trovano all'interno del computer o che lo collegano al mondo esterno; trasportano informazioni sullo stato della CPU, della RAM ed altre caratteristiche hardware. Il bus del CPC464 è il più largo dei due pettini accessibili da fori situati sul retro della tastiera.

Byte

Un gruppo di otto bit, che formano la parte più piccola della memoria che una CPU ad otto bit possa richiamare e memorizzare. Si vedano le ulteriori informazioni nell'Appendice II.

CAD

Computer Aided Design (progettazione assistita dal computer). Una interazione di potenza di calcolo e grafica al fine di fornire una tavola di disegno elettronica, sebbene qualsiasi calcolo eseguito su un computer nel corso di un progetto prende il nome di CAD.

CAE

Computer Aided Education (educazione assistita dal computer). L'aiuto del computer per l'educazione. CAI (Computer Aided Instruction ovvero istruzione assistita dal computer) e CAl (Computer Aided Learning ovvero apprendimento assistito dal computer) sono due aspetti del CAE.

Canale

La via usata dal computer per effettuare un output: lo schermo, la stampante o una cassetta.

Carattere

Un qualsiasi simbolo che può essere rappresentato e visualizzato dal computer, incluso lettere, numeri e simboli grafici. (Si veda Appendice II)

Carattere grafico

Una forma o un modello che può servire a creare immagini. Il 464 ha un set completo di caratteri grafici descritti nell'Appendice III.

Cartuccia

Uno speciale circuito integrato contenente software che può essere inserito direttamente nel computer mediante una presa specifica. Il software fornito su cartuccia viene caricato ed eseguito più velocemente delle normali cassette, ma il suo costo è notevolmente superiore.

Cassette

Oltre alle ovvie cassette di musica è un termine generico che racchiude una gamma di 'pacchetti' incluso il software ROM, ecc.

Cattura di dati

Il termine che descrive la raccolta di dati da una sorgente esterna che è collegata in qualche modo ad un computer centrale.

Chip

Un ingannevole ma popolare riferimento a qualsiasi forma di circuito integrato elettronico. Il 'chip' è in effetti un piccolo rettangolo di materiale silicio con il quale il circuito è costruito.

Ciclo

Un processo in un programma che viene eseguito ripetutamente dal computer finchè una condizione viene soddisfatta.

Circuito integrato

Un insieme di circuiti elettronici miniaturizzati ed inseriti in un unico pezzo di silicio. Vedere anche chip.

Codice

A prescindere dalle implicazioni letterali, tale termine è usato frequentemente come abbreviazione di 'codice macchina'.

Codice a barre

Si guardi ad esempio sotto la confezione di una saponetta. Un computer che legge codici a barre può leggere mediante tecniche ottiche come la scansione mediante un laser a bassa energia.

Codice macchina

Il linguaggio di programmazione direttamente comprensibile da microprocessore, i cui comandi sono composti da cifre binarie.

Comando

Una istruzione di programmazione.

Compilatore

Un programma che converte programmi scritti in un linguaggio interpretativo ad alto livello come il BASIC in istruzioni in codice del microprocessore, in tal modo le operazioni vengono eseguite ad una velocità molto maggiore.

Computer della quinta generazione

Sono soprattutto grossi computer che si pensa possano giungere ad autoprogrammarsi usando gli sviluppi dell'intelligenza artificiale.

CP/M

Sistema operativo della Digital Research per computer a 8 bit standard de facto.

CPU

Central Processing Unit (Unità centrale di elaborazione). Il componente nel cuore di qualsiasi computer che interpreta le istruzioni e fa sì che quest'ultimo le esegua; in un microcomputer, la CPU è il microprocessore stesso.

Cursore

Un indicatore spostabile che indica il luogo, sullo schemo, in cui apparirà il carattere successivo.

Cursore grafico

E' simile al cursore del testo, ma è utilizzato nello schermo grafico. E' un elemento invisibile nel CPC464 ma nonostante ciò è indispensabile per fare disegni. Non deve essere confuso con i caratteri grafici (Appendice II) che appartengono al set di caratteri e vengono stampati alla posizione del cursore del testo.

Debugging

L'eliminazione degli errori in un programma mediante una combinazione di tentativi e metodi più scientifici.

Diagnostica

Un messaggio prodotto automaticamente dal computer per indicare e identificare un errore in un programma.

Diagramma di flusso

Una rappresentazione dei passi di un programma che indica la sequenza degli eventi che avvengono durante l'esecuzione di un programma.

Digitale

Descrive l'espressione di modifica di quantità in termini di passi discreti piuttosto che continui. E' l'opposto di analogico.

Digitalizzatore

Un dispositivo per inserire informazioni analogiche in un computer. Si fa riferimento di solito in unione con le tavolette grafiche.

Disco
Un disco di plastica piatto e circolare rivestito su uno o entrambi i lati con una superficie magnetica ed usato come mezzo di memorizzazione dei dati. Il disco è sempre situato in una busta di protezione quadrata di carta o di plastica; ad esso si accede mediante una finestra di lettura situata sulla busta di protezione. Si veda anche Floppy e Winchester.

Documentazione
I manuali forniti con il computer o il software che spiega il suo uso.

DOS
Disk Operating System. Il software che controlla tutte le operazioni di un drive; si comporta come un sorvegliante nel 'parcheggio' rappresentato dalla disposizione logica del disco.

Drive
L'unità che registra le informazioni sulla superficie magnetica di un disco e 'legge' le informazioni registrate.

Editing
Correzione di dati, di un programma o di testi.

Editor
Un programma che si trova normalmente nella ROM del computer e che permette il processo di editing.

Editor di schermo
Un editor di testi o programmi in cui il cursore può essere spostato sullo schermo.

EPROM
Erasable Programmable Read Only Memory. Simile a una PROM ma i dati contenuti nel chip possono essere cancellati usando raggi ultravioletti e quindi riprogrammati. Una EEPROM è simile ma viene cancellata elettricamente.

Errore di sintassi
Si presenta se viene infranta una regola sull'uso delle parole chiave o delle variabili. Il BASIC presenterà il messaggio Syntax error. Vedere Appendice VIII.

Espressione
Una formula semplice o complessa usata in un programma per eseguire un calcolo. L'espressione definisce normalmente la natura dei dati che può utilizzare.

File

Una raccolta di informazioni, normalmente memorizzata su cassetta o disco. Alcuni computer possono usare anche file su RAM.

Firmware

Software contenuto in ROM: un punto di incontro tra puro software e puro hardware.

Floppy

Un disco magnetico estraibile di 5.25' o di 8' di diametro utilizzato per memorizzare i dati. E' contenuto in una busta protettiva; ha una capacità molto superiore a quella di una cassetta. E' molto più veloce e più costoso.

Foglio elettronico

Un programma che permette di riempire e manipolare righe e colonne di numeri. Modificando un numero cambiano tutti quelli che da esso dipendono.

Forth

Un linguaggio di programmazione ad alta velocità con una velocità e una complessità che si trova tra un linguaggio ad alto livello e il linguaggio macchina: non è un linguaggio per principianti.

Generatore sonoro

La parte hardware e software di un computer dedicata alla produzione di suoni.

Generazioni di computer

Le nuove scoperte tecnologiche hanno prodotto diverse fasi di sviluppo nel campo dei computer. In tal modo si sono prodotte diverse generazioni di computer a seconda delle varie tecnologie usate per costruirli.

Giochi di avventura

Un culto per alcuni e una noia per altri. Giochi basati su testi nei quali l'utente che gioca è invitato a partecipare in una serie di eventi pseudo-casuali basati sulla ricerca di una via di uscita in un labirinto.

Giochi d'azione

E' il tipo di gioco per computer dove: gli uomini spaziali invadono e vengono conquistati, mostri insaziabili inseguono in un labirinto e si impadroniscono di un incauto, personaggi sotto il controllo dell'utente cercano di evitare in tutti i modi una spiacevole 'morte'. Sono divertenti e permettono di sviluppare buoni riflessi ma sono di scarsa utilità per gli studenti informatici.

Grafica

La parte dello schemo del computer che non è in relazione alla visualizzazione dei caratteri ma al disegno di linee, di cerchi, di grafici ecc. Con una stampante appropriata è possibile anche avere copie dello schermo su carta.

Handshaking

Una sequenza di segnali elettronici che inizia, controlla e sincronizza lo scambio di dati tra un computer e una periferica o tra due computer.

Hardcopy

Stampa su carta di un programma, di altri testi o di uno schermo grafico. In questo caso lo schermo viene chiamato softcopy.

Hardware

La parte elettronica e meccanica di un computer; tutto ciò che non è software o firmware.

I/O

Input/Output.

IEEE-488

Una delle interfacce standard per collegare i dispositivi ad un microcomputer. E' simile ma non completamente compatibile con l'interfaccia parallela Centronics.

Indirizzo

E' il numero in una istruzione che identifica la locazione di una 'cella' nella memoria del computer. Mediante tale indirizzo è possibile selezionare una locazione di memoria particolare in modo sia possibile trovare e leggere il suo contenuto o, nel caso della RAM leggervi e scrivervi dopo eventuali modifiche.

Ingegneria del software

Indica la programmazione di un computer, utilizzando tecniche precise e non un approccio intuitivo.

Inizializzazione

Accensione di un sistema e dichiarazione di particolari valori delle variabili prima di iniziare l'esecuzione di un corpo di un programma; ad esempio la dichiarazione di una variabile intera, ecc.

Input

Tutto ciò che entra nella memoria del computer dalla sua tastiera, dalla sua unità a cassette, dal suo disco, dall'interfaccia seriale o da un altro dispositivo.

Insieme di istruzioni

Il processo logico e matematico più importante eseguito da un microprocessore. Ogni istruzione di alto livello (compresi i codici mnemonici dell'assembler) devono essere tramutati in istruzioni comprensibili dal computer. Un solo comando ad alto livello può provocare l'esecuzione di molte istruzioni dell'insieme compreso dalla CPU.

Intelligenza artificiale AI

Tecniche di programmazione che permettono al programma di apprendere dalle esperienze passate.

Interattivo

Si riferisce normalmente a programmi in cui il computer chiede all'utente di fornire vari tipi di input che possono servire a controllare una nave spaziale in un gioco di azione o a rispondere a domande in un programma educativo. L'azione dell'utente avrà effetto in tempo reale, cioè durante l'esecuzione del programma.

Interfaccia

E' una porta di accesso al computer sia in termini elettrici che umani. Le interfacce di cui è dotato il CPC464 sono la tastiera (input), lo schermo (output) e tutte le prese che si trovano sul retro e che servono a collegare varie periferiche al computer.

Interfaccia parallela

Interfaccia per stampante del CPC464 che permette di utilizzare una stampante parallela; ciò significa che ogni linea di dati del bus viene collegata ad un corrispondente input della stampante. I dati vengono trasferiti molto più velocemente usando un'interfaccia parallela piuttosto che una interfaccia seriale in quanto quest'ultima deve prima preparare il formato di un byte ed inserire poi informazioni di sincronizzazione.

Interfaccia Seriale

Questo termine si riferisce soprattutto all'interfaccia RS232 ma esistono altri standard per la comunicazione seriale dei dati.

Interprete

La parte del software di sistema interpreta un linguaggio ad alto livello per fare in modo che possa essere compreso dalla CPU. Ad esempio il codice BASIC immesso da tastiera viene convertito nel linguaggio interno del computer.

Istruzione

Un comando che chiede al computer di eseguire una data operazione. Una sequenza di istruzioni formano un programma.

Iterazione

Uno degli elementi del calcolo. Il computer esegue tutte le operazioni spezzandole in parti più semplici che possono essere gestite dalla CPU. Per fare ciò il computer deve continuare ad eseguire molti elementi più semplici fino al verificarsi di una particolare condizione.

Joystick

Un dispositivo di input che sostituisce generalmente le funzioni dei tasti cursore e rende molti giochi più facili e più semplici.

K

Abbreviazione di Kilo che nel mondo dei calcolatori è molto utilizzato per kilobyte che corrisponde a 1024 byte e cioè a 2 elevato a 10. Vedere Appendice II.

Leggibile da una macchina

Un mezzo o una informazione che può essere direttamente compresa da un computer senza immissione dalla tastiera.

Linguaggi di alto livello

Linguaggi scritti in una forma facilmente comprensibile in cui il vero linguaggio fa la maggior parte del lavoro di interpretazione. Sono più lenti dei programmi in linguaggio macchina ma molto più comprensibili.

Linguaggio a basso livello

Ad esempio il linguaggio Assembler. E' un linguaggio di programmazione in cui ogni istruzione corrisponde ad una istruzione della macchina.

Linguaggio di programmazione

E' ciò che permette di scrivere un programma; è costituito da un insieme di rigide regole sull'uso delle parole e dei numeri.

LISP

Abbreviazione di LISt Processing language o linguaggio per la gestione delle liste. E' un altro linguaggio ad alto livello per computer.

Logica

Componenti elettronici che eseguono le elementari operazioni e funzioni logiche che compongono ogni operazione di un computer.

Logo

E' un linguaggio per computer semplice da imparare ed orientato alla grafica usato frequentemente nelle scuole come aiuto all'insegnamento dell'informatica.

LSI

Large Scale Integration (Integrazione spinta a larga scala). Lo sviluppo dei circuiti integrati ha permesso di inserire molte funzioni in frammenti di silicio sempre più piccoli.

Mappa di memoria

La disposizione della memoria con i vari indirizzi e la locazione di memoria delle varie funzioni, ad esempio lo schermo, il sistema di gestione del nastro, ecc.

Matrice

La disposizione di punti che formano la cella di carattere sullo schermo o sulla testina di stampa di una stampante a matrice. E' anche un termine usato in matematica e in informatica per denotare i vettori.

Memoria

Il computer memorizza molte informazioni e dati disposte in modo logico e accessibili uno ad uno. E' anche conosciuto come RAM, memoria ad accesso diretto, dove le informazioni possono essere memorizzate o lette, o ROM dove le informazioni possono essere lette ma non scritte. I dischi e i nastri sono esempi di memoria esterna.

Menù

Un elenco delle varie opzioni che possono essere eseguite in un determinato ambito e che possono essere selezionate dall'utente.

Microprocessore

Un circuito integrato che si trova nel cuore del computer e che esegue le istruzioni che gli vengono presentate dall'interprete BASIC.

Modem

Un MODulatore/DEMulatore che permette di collegare il computer ad una linea telefonica o a un altro dispositivo di trasmissione seriale comprese le fibre ottiche. Vedere anche accoppiatore acustico.

Modo grafico

I primi microcomputer dovevano essere predisposti appositamente per gestire o i caratteri o la grafica. I moderni personal computer possono utilizzare simultaneamente testo e grafica.

Modulatore RF

Un dispositivo che permette di inviare l'output video di un computer alla presa d'antenna di un normale televisore.

Monitor

Lo schermo di un computer; il termine descrive anche il programma in linguaggio macchina che permette di accedere alle operazioni più a basso livello del computer.

Mouse

Una piccola scatola che contiene una sfera di gomma. Spostato su una superficie permette di muovere il cursore a piacimento. E' stato progettato per superare la paura della tastiera e per rendere il software più amichevole.

Nibble

Mezzo byte. Una espressione binaria a quattro bit. Ogni cifra di un numero esadecimale rappresenta un nibble.

Notazione decimale

Sistema di numerazione in base 10, nel quale si usano cifre da 0 a 9, che rappresentano unità, decine, centinaia, migliaia, ecc.

Notazione esadecimale

Sono numeri in base sedici. Vedere Appendice II. Indicati nel CPC464 da prefisso & o &H.

Notazione polacca inversa

E' un metodo utilizzato da alcuni calcolatori per eseguire le operazioni aritmetiche. In tale notazione, gli operatori (+, -, *, /) vengono posti di fronte ai valori ai quali vengono applicati.

Numeri a virgola fissa

Un numero rappresentato, utilizzato e memorizzato con un punto decimale in posizione fissata.

Numeri a virgola mobile

Un numero reale utilizzato e memorizzato con un punto decimale che può trovarsi nella posizione desiderata. Questo metodo è particolarmente utile quando si utilizzano grossi numeri.

Numeri binari

Un numero rappresentato in notazione binaria. Denotati nella programmazione con il CPC464 dal prefisso &X; ad esempio: &X0101 è equivalente (in decimale) a 5.

Numero casuale

Un numero generato dal computer in modo irripetibile ed imprevedibile.

Numero da leggere bit a bit
In tali numeri l'informazione viene estratta considerando lo stato di ognuno degli otto bit che compongono un byte. Il valore decimale intero può non avere significato.

Numero di linee
Il BASIC e alcuni altri linguaggi usano programmi il cui ordine è stabilito dai numeri di linea.

Numero intero
Un numero senza parte frazionale, cioè un numero senza virgola. E' il contrario del numero reale che è composto da una parte intera e una parte frazionale.

Numero reale
Un numero che ha una parte intera ed una frazionale. Una variabile reale rimane tale anche se in un dato momento dell'esecuzione del programma può contenere un numero intero.

Off line
La periferica di un computer (un terminale o una stampante) che non è connessa in modo attivo o utilizzabile da l'unità di elaborazione.

On line
L'opposto di off line.

Operatore
La parte di un'espressione aritmetica che permette di modificare dei numeri, ad esempio +, -, / ecc.

Orologio
Il sistema di temporizzazione di un computer usato per sincronizzare e gestire le operazioni di un computer. L'orologio in tempo reale mantiene l'ora, la data ecc.

Ottale
Un sistema numerico in base otto in cui ogni cifra (da 0 a 7) è composto da tre bit.

Output
Tutto quanto esce da un computer come risultato di qualche calcolo.

Paddle
Nome alternativo per il joystick.

Parola chiave

Una parola il cui uso all'interno di un programma o di un linguaggio è riservato per una particolare funzione o comando.

Parola riservata

Una parola che ha un significato particolare all'interno di un programma e che non può perciò essere utilizzata per altri scopi. Ad esempio, il BASIC non potrà accettare NEW come nome di variabile in quanto essa è già riservata per altri scopi.

Pascal

Un linguaggio di programmazione strutturato ad alto livello che deve essere compilato prima di essere eseguito e che quindi viene eseguito molto velocemente. E' in genere il linguaggio di programmazione, dopo il BASIC, che uno studente apprende.

PEEK

La funzione BASIC che permette di leggere direttamente la memoria del computer e restituisce il valore di una data locazione.

Penna ottica

Un dispositivo alternativo di input che utilizza una penna.

Periferica

Stampanti, Modem, Joystick, drive per dischi - tutto quanto si collega al computer per espandere le sue capacità.

Pixel

La più piccola parte di schermo che può essere controllata dall'hardware.

Plotter

Un particolare tipo di stampante che permette di tracciare disegni utilizzando delle penne invece di una testina di stampa. E' usato per output tecnico e grafico.

POKE

La parola chiave del BASIC che permette di inserire un dato valore in una specifica locazione di memoria - vedere Capitolo 8.

Porta

Le porte logiche permettono il passaggio di dati al verificarsi di determinate condizioni. Vi sono diversi tipi di porte (OR, AND, XOR, ecc.). Si veda Algebra booleana.
Un punto di collegamento per l'input e l'output dei dati.

Portabilità

Indica la possibilità di utilizzare un determinato programma su vari computer - ciò è permesso da sistemi operativi compatibili come CP/M della Digital Research.

Programma

E' l'insieme di istruzioni che fa risolvere al computer un determinato problema; può andare da una semplice routine in codice macchina ad un intero programma di elaborazione di testi.

Programma di avviamento

I programmi e sistemi operativi non si caricano da soli ma vengono caricati da una piccola routine situata (di solito) nella ROM che effettua il processo di caricamento in una locazione di memoria specifica.

Programmazione strutturata

Una tecnica di programmazione che permette di produrre programmi che seguono passi ben definiti.

Programmi applicativi

E' un programma che invece di essere uno strumento di applicazione generale ha un compito preciso ad esempio un assemblatore, un driver per stampanti ecc...

PROM

Programmable Read Only Memory. Un circuito integrato che una volta scritto non può essere cancellato (vedere anche EPROM).

Prompt

Simbolo o messaggio visualizzato sullo schermo che permette all'utente di rispondere al computer o continuare un input. Il BASIC usa un semplice punto di domanda per richiedere dell'input o la parola Ready quando attende un comando.

QWERTY

Un termine di uso comune che descrive una tastiera inglese.

RAM

Random Access Memory o memoria ad accesso diretto. E' un tipo di memoria che può essere letto o scritto, mediante i circuiti del computer in fase di esecuzione del programma.

Raster

Un tipo di scrittura sullo schermo, in cui le immagini sono composte da un certo numero di scansioni orizzontali.

Refresh
Permette di aggiornare le informazioni, sullo schermo o in memoria. Non è necessariamente un processo distruttivo, a volte ripristina semplicemente ciò che è già presente sullo schermo o in memoria.

Registro
Una locazione di memoria all'interno della CPU.

REM
Una istruzione che non viene eseguita ma serve a ricordare al programmatore che cosa il programma sta facendo e quando il programma è stato fatto.

Rete
Quando due o più computer sono collegati insieme per scambiare dati e informazioni o via cavo o via modem.

Riconoscimento ottico dei caratteri
Un mezzo per leggere caratteri stampati o scritti utilizzando un lettore ottico.

Riconoscimento vocale
La conversione delle parole in istruzioni leggibili da una macchina.

Ricorsione
Un insieme di passi all'interno di un programma o di una routine in cui il risultato di ogni ciclo è in relazione con il precedente.

Risoluzione
La capacità di stabilire dove termina un elemento dello schermo e dove inizia il successivo. Indica anche la possibilità del computer di eseguire operazioni su numeri molto grossi.

ROM
Read Only Memory, memoria a sola lettura, una volta scritta non può essere né cancellata, né riscritta.

Routine
Parte del programma che esegue delle operazioni ben determinate. Può trovarsi all'interno del programma o al suo esterno, in modo che possa essere incorporata. Un programma che genera un orologio può essere considerato una routine.

RS232C
Uno standard per le comunicazioni seriali. Entrambi i dispositivi che si trovano alle estremità devono essere configurati secondo le caratteristiche della trasmissione. Si confronti con l'interfaccia Centronics.

Rumore
Le possibilità sonore del CPC464 permettono di inserire una quantità variabile di rumore usando il comando sound in modo da creare effetti come ad esempio delle esplosioni.

Scorrimento
E' il modo in cui lo schermo scorre quando viene raggiunto il fondo.

Separatore
Esegue la stessa funzione di un delimitatore: indica i confini tra le parole riservate ed altri elementi di un programma o dei dati.

Set di caratteri
Tutte le lettere, numeri e simboli disponibili su un computer o su una stampante. Il fatto che un carattere esista su un computer non implica necessariamente che questo sia disponibile su una stampante.

Simulazione
Tecnica per sperimentare su computer alcuni processi della vita reale, ad esempio il volo, la guida di un'auto, ecc.

Sintesi vocale
Generazione di parole usando hardware e software.

Sistema operativo
Il sorvegliante del "parcheggio" costituito dalla memoria del computer. E' composto da software che gestisce le precedenze e le sincronizzazioni nelle operazioni del computer.

Software
Vedere Programmi.

Sovrascrittura
Cancellazione di un'area di memoria e sostituzione del suo contenuto con altri dati.

Sprite
Un carattere dello schermo che si può muovere liberamente per lo schermo.

Stack
Un'area di memoria che serve ad "impilare" informazioni, ma in cui solo l'ultimo elemento della fila può essere richiamato.

Stampante
Dispositivo che permette di stampare testi e grafica.

Stampante a margherita
Una stampante che può produrre documenti di alta qualità. I caratteri stampati vengono creati mediante l'impatto delle lettere contro il nastro inchiostrato.

Stringa di caratteri
Una parte di dato <variabile> comprendente una sequenza di caratteri che può essere memorizzata o trattata come una singola unità, cioè una parola o un insieme di parole.

Subroutine
Vedere Routine.

Tabelle di verità
Il risultato di un'operazione logica può essere vero o valso, 1 o 0. Le tabelle di verità indicano i valori di verità del risultato di un'operazione logica.

Tasti di controllo del cursore
Tasti che spostano il cursore nello schermo e che sono frequentemente usati per controllare la direzione di azione nei giochi; sono indicati da frecce.

Tasti funzione
Ad essi è stato assegnato un compito specifico che si può aggiungere o sostituire a quanto è scritto sopra il tasto. Alcuni tasti del CPC464 possono essere definiti come tasti funzione, nel qual caso la pressione di un singolo tasto può provocare l'esecuzione di più istruzioni.

Tastiera
L'insieme dei tasti alfanumerici di cui il computer è dotato per permettere l'inserimento di comandi e altre informazioni.

Tastierino numerico
L'area sulla tastiera dove si trovano i tasti numerici in modo da facilitare l'inserimento dei dati numerici. Nel caso del CPC464 vi si trovano anche i tasti funzione definibili dall'utente.

Tavoletta grafica
Un dispositivo che permette di prelevare le coordinate dei punti di un certo disegno per manipolarlo all'interno del computer. E' una forma di conversione analogica digitale.

Tecnologia dell'informazione
Tutto ciò che è in relazione con l'uso dell'elettronica nell'elaborazione dell'informazione e nelle comunicazioni: l'elaborazione di testi, le comunicazioni di dati, Videotel ecc.

Terminale
Una tastiera con uno schermo o una stampante che permette di collegarsi ad un computer.

Terminale generico
Un terminale per computer che si comporta semplicemente come un dispositivo di input ed output e non elabora in alcun modo le informazioni. Un terminale stupido è invece un terminale in cui non esiste nessun circuito per la gestione del video ed in cui le informazioni vengono ricevute semplicemente con un segnale video.

Troncamento
Un numero o stringa a cui è stata tolta la testa o la coda.

Utility
Un programma che esegue una operazione specifica, ad esempio l'ordinamento o la copia di un file.

Variabile
Un oggetto contenuto in un programma che può essere richiamato mediante il nome ma il cui valore può cambiare durante l'esecuzione del programma.

Appendice II

Basi dell'uso di un calcolatore

Chi ha paura del "gergo"?

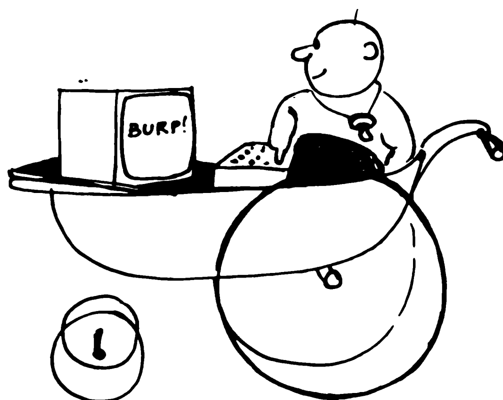
Così come tutte le discipline, il mondo dei calcolatori ha sviluppato un proprio linguaggio come forma di abbreviazione per concetti complicati che richiedono più parole per essere spiegati. Non è solo l'alta tecnologia colpevole di nascondersi dietro una apparente cortina di fumo o 'ronzii di parole', gergo e terminologia; quanti di noi si sono scontrati con le barriere della comprensione, che esistono in tutti i campi della vita?

La maggiore differenza tra i gerghi sorge dal modo con cui le parole vengono usate piuttosto che dal loro significato. La maggior parte delle persone che diventano familiari con la terminologia del calcolo adotterà un modo personale di uso delle parole che sarà più diretto possibile, al fine di minimizzare la complessità della comunicazione.

Non bisogna sentirsi ingannati da tale 'linguaggio chiaro' usato nel calcolo; non è una disciplina letteraria ma una scienza precisa, separata dalla 'sintassi' delle parole; la struttura della comunicazione è molto diretta, e non confusa o ambigua. Gli insegnanti di informatica non hanno cercato di rendere ciò una forma d'arte cercando di analizzare il significato esatto di una parola secondo ciò che intende un programmatore nella costruzione di un programma.

Avendo detto ciò, sia che il significato di programma sia o meno ovvio, vi sono ancora diversi aspetti che possono essere analizzati sia in modo elegante che disordinato, ed è stata data più importanza ad un approccio formale nella costruzione di un programma ora che il danno iniziale portato dalla micro rivoluzione è stabilizzato.

Il calcolo viene appreso rapidamente da molte persone giovani che apprezzano la precisione e la semplicità delle idee ed il modo con cui queste possono essere comunicate; non si troveranno molti ragazzi di dieci anni avvocati, ma sarà possibile trovare molti programmatori!



Le basi del BASIC

Quasi tutti i computer domestici forniscono un linguaggio noto come BASIC il quale permette di ottenere programmi scritti nel modo migliore e con un linguaggio semplice e subito disponibile. BASIC è un abbreviazione di 'Beginners All-purpose Symbolic Instruction Code': molti programmi estremamente complessi e potenti sono scritti usando BASIC.

Comunque non c'è dubbio che il nome ha attratto molti principianti per la promessa di fornire un punto di partenza nel labirinto dei linguaggi di programmazione e ha contribuito notevolmente alla sua divulgazione.

Da ora in poi, le parole che si usano comunemente e che formano il glossario dei termini del calcolo verranno introdotte per la prima volta scrivendole in corsivo. Non occorre preoccuparsi se non si conosce il significato di tali parole, c'è un indice di queste nel glossario.

Le basi

BASIC è un linguaggio di programmazione che interpreta un certo numero di comandi ed esegue le operazioni sui dati in fase di esecuzione del programma. Diversamente dal vocabolario umano che comprende dalle 5000 alle 8000 parole (oltre alle varie desinenze dei verbi che si possono usare, ecc) il BASIC ne possiede duecento. I programmi scritti usando BASIC devono seguire regole rigide riguardanti l'uso di tali parole. La sintassi è precisa e qualsiasi tentativo di comunicazione con il computer mediante l'uso di espressioni letterali o colloquiali darà come risultato un freddo messaggio:

```
Syntax error
```

ovvero errore di sintassi.

Ciò non è restrittivo come potrebbe sembrare ad un primo momento poichè il linguaggio BASIC (la sua sintassi) è progettato prima di tutto per trattare numeri, dati numerici. Le parole sono essenzialmente una estensione dei familiari operatori aritmetici */- ecc, e il concetto più importante da comprendere dai principianti è quello che un computer lavora solo con un dato numerico: ogni informazione è fornita al Processore Centrale (circuito integrato) solo sotto forma di dati numerici.

NUMERI PER FAVORE!

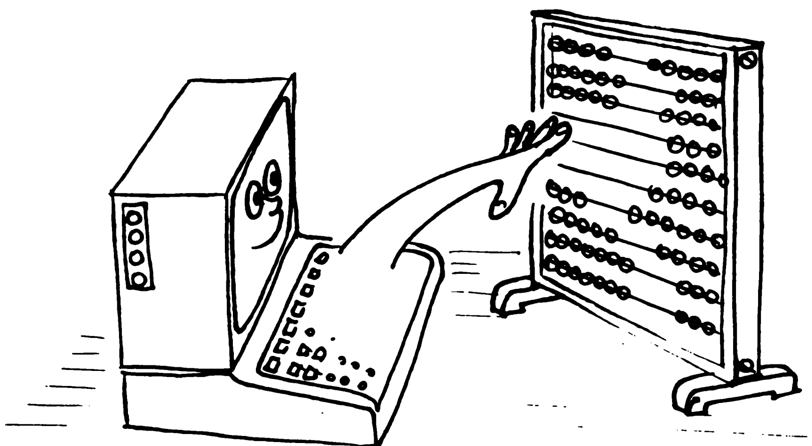
Se un computer viene usato per memorizzare tutto il lavoro di Shakespeare, nel sistema non si troverà una sola lettera o parola. Ogni informazione viene prima convertita in un numero che il computer può localizzare e successivamente trattare come richiesto.

BASIC interpreta le parole come numeri che il computer può successivamente trattare usando solo l'addizione, la sottrazione ed alcune caratteristiche Booleane che permettono di confrontare dati e selezionarne altri in base a determinati attributi: cioè controllare se un numero sia è più grande o uguale ad un altro, o eseguire un particolare compito se un nunero O un altro incontra una certa scelta.

Mediante il programma il computer suddivide ogni compito in una semplice serie di operazioni che implicano due sole risposte : SI o NO.

Il processo di moltiplicazione viene eseguito usando diverse addizioni: l'istruzione BASIC che moltiplica 35 per 10 (35*10) si ottiene addizionando 35 a se stesso dieci volte.

Parte del Central Processor Unit (CPU o Unità centrale) è caricata con il dato numerico 10 ed un'altra parte della CPU con 35. 35 viene aggiunto ogni volta a se stesso e la parte di memoria contenente 10 viene decrementata di uno fino a quando non raggiunge valore zero; a tal punto il processo termina ed il risultato accumulato, 350, viene inviato in un'altra parte della CPU come Output.



Che tale processo sembri scomodo non è del tutto sbagliato, poichè si è appena scoperto una delle più importanti verità riguardo il calcolo. Un computer è prima di tutto uno strumento per l'esecuzione di compiti semplici e ripetitivi in modo veloce e con precisione assoluta. Così BASIC interpreta le istruzioni fornite sotto forma di programma e le traduce nel linguaggio che può essere usato dalla CPU. Solo due stati vengono compresi dalla logica del computer: 'si' o 'no' rappresentati in notazione binaria da '1' e '0'. La rappresentazione Booleana è semplicemente 'vero' e 'falso', non vi sono espressioni come 'possibile' o 'forse'.

Il processo di commutazione tra questi due stati distinti è l'essenza del termine digitale. Nella natura la maggior parte dei processi si muovono gradualmente da uno stato completamente 'stabile' in un altro in una progressione omogenea. In altre parole la transizione è fatta seguendo un percorso lineare tra due stati, in un ambiente digitale ideale la commutazione da uno stato al seguente viene fatto istantaneamente, ma la fisica dei semiconduttori provoca un ritardo a cui si fa riferimento come ritardo di propagazione ed è l'accumulazione di molti di questi ritardi la ragione per cui il computer impiega del tempo per fornire una informazione.

In qualsiasi caso, il computer deve attendere che una operazione venga finita prima che la succesiva possa usare il risultato della prima: quindi ci dovrebbe essere comunque un ritardo imposto. Il processo digitale è nero o bianco dove i periodi nella transizione mediante sfumature di grigi non hanno significato. La progressione utilizza varie sfumature di grigio.

Se in definitiva la risposta è 0 o 1, non vi è possibilità che sia 'quasi' corretta. In effetti può sembrare a volte che il computer faccia degli errori nel trattamento dei dati numerici a causa della limitazione della dimensione dei numeri che esso può utilizzare; in altre parole, numeri molto grossi o molto piccoli dovranno essere ristretti all'interno dello spazio standard che è stato loro assegnato e ciò porta ad errori di arrotondamento per i quali 999.999.999 può diventare 1.000.000.000.

In altre parole dove vi sono solo due numeri disponibili, 0 e 1, come contare dopo l'uno?

Bit e byte

Abbiamo appena accennato all'uso dei numeri basati sul sistema decimale nel quale il punto di riferimento è 10, vi sono cioè dieci cifre disponibili per rappresentare quantità le quali vanno da 0 a 9 (che sono prevalentemente usate da 1 a 10). Il sistema in cui la gamma di numeri va da 0 a 1 è il sistema binario e le cifre del sistema sono dette bit, una abbreviazione di 'BInary digIT' (cifra binaria). Non si rischierà di fare confusione se si dimentica del tutto il contesto decimale di 0 e 1 e si usano due simboli completamente diversi per rappresentare una funzione: fl e*.

La relazione tra bit e notazione decimale è semplice da comprendere:

E' convenzione attuale dichiarare il numero massimo di cifre binarie usate aggiungendo all'inizio del numero alcuni zeri per terminare il numero:

7 decimale diventa
00111 binario
usando 5 bit di notazione

Nel sistema binario, le cifre possono essere considerate semplicemente come indicatori in colonne per specificare se deve o meno venir fornita la potenza di due se presente (1) o no (0).

$2^0 = 1$
$2^1 = 2 = 2 = 2(2^0)$
$2^2 = 4 = 2x2 = 2(2^1)$
$2^3 = 8 = 2x2x2 = 2(2^2)$
$2^4 = 16 = 2x2x2x2 = 2(2^3)$

quindi la colonna apparirà

2^4	2^3	2^2	2^1	2^0
1	0	0	1	1
(16	0	0	2	1) = 19

Al fine di fornire un metodo per abbreviare il riferimento ad una informazione binaria, viene usato il termine byte che denota 8 bit di informazione. Il numero più alto che può essere memorizzato in un byte è (binario) 11111111 o (decimale) 255. Ciò implica 256 variazioni incluso 00000000, che è ancora un dato valido per un computer.

I computer tendono a trattare dati in multipli di 8 bit. 256 non è un numero molto grande quindi, per ottenere un accettabile quantità di memoria, si usano due byte al fine di fornire un metodo di indirizzamento della memoria sotto forma di vettore, con un indirizzo orizzontale ed uno verticale mediante il quale gli elementi del vettore possono essere localizzati:

0	1	2	3	4	5	6	7	8	9
1									
2									
3									
4									
5			1		1				
6									
7									
8									
9									

Questo vettore può contenere (10x10) oggetti di informazione usando i numeri di indirizzo compresi nel campo 1 - 9. L'oggetto memorizzato nella posizione 3,5 è '1', così come l'oggetto nella posizione 5,5.

Quindi un vettore binario di 256x256 può trattare 65.536 locazioni individuali usando un indirizzo ad 8 bit per gli assi verticali e orizzontali del vettore. Quindi il nostro '0' e il nostro '1' si sono progrediti rendendosi in grado di identificare uno dei 65.536 elementi.

Il livello successivo di abbreviazione in notazione binaria è kilobyte (kByte o semplicemente 'K') il quale è 1024 byte. 1024 è il multiplo binario più vicino alla nostra notazione di 'chilo' e spiega perchè un computer descritto avente '64K' di memoria ha in effetti una memoria di 65.536 byte (64x1024).

Grazie all'interprete BASIC non è necessario effettuare conversioni ed è possibile essere dei bravi programmatori senza conoscere il sistema binario. Sebbene una valutazione del significato binario aiuterà a distinguere i molti 'magici' o significativi numeri che inevitabilmente salteranno fuori nel corso del lavoro.

E' bene apprendere il sistema binario e i vari significati dei numeri 255, 1024, ecc. poichè è molto improbabile che questi cambieranno in futuro. La certezza e la semplicità derivante dal lavoro con solo due stati prevarrà notevolmente sulla complessità derivante dall'uso di altri numeri base.

Comunque...

Semplice o elegante che sia, il sistema binario è prolisso e comporta imprecisioni poichè non può esssere letto facilmente con una occhiata. Il codice binario ha un certo numero di codici associati che agiscono come abbreviazione per i programmatori. Uno di tali codici, ampiamente usato nei microcomputer, è detto Hex (abbreviazione di esadecimale).

Il numero è basato sul 16 ed è rappresentato da un solo carattere:

Decimale

0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15

Hex

0 1 2 3 4 5 6 7 8 9 A B C D E F

Il sistema esadecimale può suddividere gli otto bit di un byte in quattro bit poichè 15 è il numero a quattro bit: 1111 binario. Il primo blocco indica il numero di unità di '15' complete, il secondo indica il resto; è in questo caso che emerge l'eleganza del sistema binario.

Riconsideriamo la tabella introdotta nella notazione binaria.

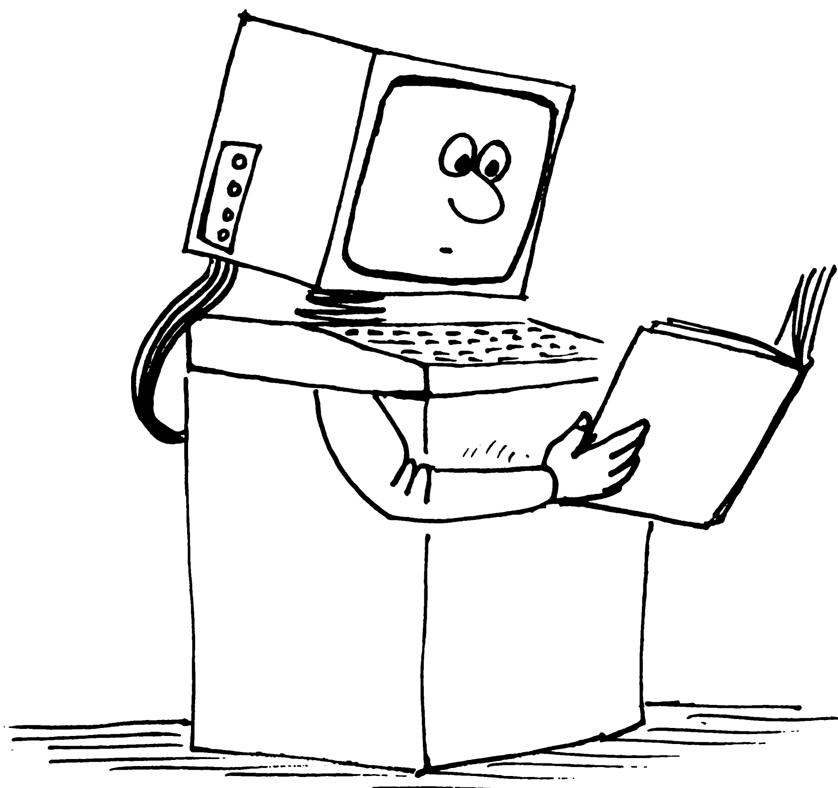
Decimale	CPC-464-ese	Binario	Esadecimale
0	fl	0	0
1	*	1	1
2	*fl	10	2
3	**	11	3
4	*flfl	100	4
5	*fl*	101	5
6	**fl	110	6
7	***	111	7
8	*flflfl	1000	8
9	*flfl*	1001	9
10	*fl*fl	1010	A
11	*fl**	1011	B
12	**flfl	1100	C
13	**fl*	1101	D
14	***fl	1110	E
15	****	1111	F
16	*flflflfl	10000	10

Il numero a 8 bit 11010110 può essere suddiviso e considerato come due numeri a 4 bit noti come nibbles, Hex D6. In questa guida un numero basato su notazione esadecimale verrà introdotto con un simbolo '&' cioè &D6, questo è il numero maggiormente usato dai programmatori che usano tecniche di linguaggio assembler. Un programma in tale linguaggio è più vicino alla programmazione in codice macchina in quanto tale linguaggio permette di usare semplici lettere 'mnemoniche' per specificare i 'numeri' effettivi di codice macchina.

Quando si usano numeri esadecimali, occorre prima di tutto calcolare il valore della prima cifra per ottenere il numero di "16" contenuti nel numero finale, e successivamente aggiungere la seconda 'metà' della notazione esadecimale per ottenere il numero decimale equivalente. Si sarà tentati di considerare di considerare il numero &D6 come 13+6, o 136. Ma è (13x16)+6=214

E' lo stesso processo che si usa quando si legge un numero decimale come '89', ovvero (8x10)+(9). Moltiplicare per dieci è notevolmente più semplice a meno che non si è fatta molta pratica moltiplicando per 16.

Se si è appreso tutto ciò senza fare troppa confusione allora si è in grado di apprendere le basi del calcolo. Ci si potrebbe essere resi conto che tutto lo scalpore è qui: questo non è del tutto sbagliato. Il computer è un gestore che dirige in modo semplice concetti ed idee: si possono eseguire compiti ad una velocità elevata (milioni di volte per secondo) e con una smisurata capacità di ricordare sia i dati inseriti che i risultati intermedi delle migliaia di semplici somme che producono il risultato. Se si desidera imparare di più sulla teoria del mondo dei computer vi sono centinaia di libri disponibili su tale soggetto. Alcuni tendono a provocare più confusione di quella che si aveva all'inizio di tale lettura e pochi rivelano realmente la semplicità e la relazione esistente tra i sistemi numerici ed il modo con cui il computer li tratta.



Appendice III

Caratteri ASCII e grafici del CPC464

III.1 ASCII

La tabella riportata illustra il normale insieme di caratteri ASCII usando la notazione decimale, ottale ed esadecimale, oltre ai codici ASCII dove sia il caso. Ognuna delle celle dei caratteri del 464 viene rappresentata in dettaglio nelle pagine seguenti.

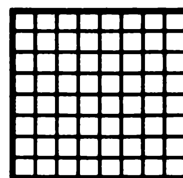
DEC	OTT	ESA	Caratteri ASCII
0	000	00	NUL ([CTRL]@)
1	001	01	SOH ([CTRL]A)
2	002	02	STX ([CTRL]B)
3	003	03	ETX ([CTRL]C)
4	004	04	EOT ([CTRL]D)
5	005	05	ENQ ([CTRL]E)
6	006	06	ACK ([CTRL]F)
7	007	07	BEL ([CTRL]G)
8	010	08	BS ([CTRL]H)
9	011	09	HT ([CTRL]I)
10	012	OA	LF ([CTRL]J)
11	013	OB	VT ([CTRL]K)
12	014	OC	FF ([CTRL]L)
13	015	OD	CR ([CTRL]M)
14	016	OE	SO ([CTRL]N)
15	017	OF	SI ([CTRL]O)
16	020	10	DLE ([CTRL]P)
17	021	11	DC1 ([CTRL]Q)
18	022	12	DC2 ([CTRL]R)
19	023	13	DC3 ([CTRL]S)
20	024	14	DC4 ([CTRL]T)
21	025	15	NAK ([CTRL]U)
22	026	16	SYN ([CTRL]V)
23	027	17	ETB ([CTRL]W)
24	030	18	CAN ([CTRL]X)
25	031	19	EM ([CTRL]Y)
26	032	1A	SUB ([CTRL]Z)
27	033	1B	ESC
28	034	1C	FS
29	035	1D	GS
30	036	1E	RS
31	037	1F	US
32	040	20	SP
33	041	21	!
34	042	22	"
35	043	23	#
36	044	24	$
37	045	25	%
38	046	26	&
39	047	27	'
40	050	28	(
41	051	29	)
42	052	2A	*
43	053	2B	+
44	054	2C	,
45	055	2D	-
46	056	2E	.
47	057	2F	/
48	060	30	Ø
49	061	31	1

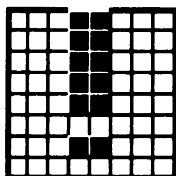
DEC	OTT	ESA	ASCII
50	062	32	2
51	063	33	3
52	064	34	4
53	065	35	5
54	066	36	6
55	067	37	7
56	070	38	8
57	071	39	9
58	072	3A	:
59	073	3B	;
60	074	3C	<
61	075	3D	=
62	076	3E	>
63	077	3F	?
64	100	40	@
65	101	41	A
66	102	42	B
67	103	43	C
68	104	44	D
69	105	45	E
70	106	46	F
71	107	47	G
72	110	48	H
73	111	49	I
74	112	4A	J
75	113	4B	K
76	114	4C	L
77	115	4D	M
78	116	4E	N
79	117	4F	O
80	120	50	P
81	121	51	Q
82	122	52	R
83	123	53	S
84	124	54	T
85	125	55	U
86	126	56	V
87	127	57	W
88	130	58	X
89	131	59	Y
90	132	5A	Z
91	133	5B	[
92	134	5C	\
93	135	5D	]
94	136	5E	↑
95	137	5F	–
96	140	60	`
97	141	61	a
98	142	62	b
99	143	63	c

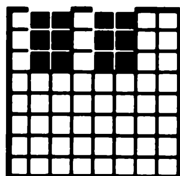
DEC	OTT	ESA	ASCII
100	144	64	d
101	145	65	e
102	146	66	f
103	147	67	g
104	150	68	h
105	151	69	i
106	152	6A	j
107	153	6B	k
108	154	6C	l
109	155	6D	m
110	156	6E	n
111	157	6F	o
112	160	70	p
113	161	71	q
114	162	72	r
115	163	73	s
116	164	74	t
117	165	75	u
118	166	76	v
119	167	77	w
120	170	78	x
121	171	79	y
122	172	7A	z
123	173	7B	{
124	174	7C	|
125	175	7D	}
126	176	7E	-

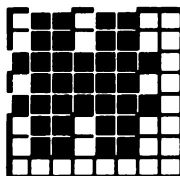
Set di caratteri grafici specifici per la macchina

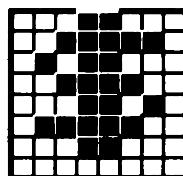
I caratteri qui riprodotti vengono riportati su una normale matrice 8 x 8 usata per la scrittura sullo schermo del 464. I caratteri definibili dall'utente possono essere riuniti per otenere speciali effetti, ponendoli uno in fianco all'altro. Vedere la descrizione fornita nel Capitolo 8.

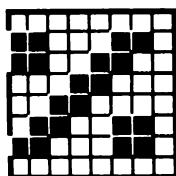

32 &H20
&X00100000

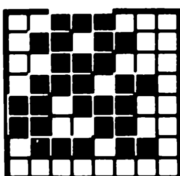

33
&H21
&X00100001

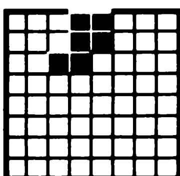

34
&H22
&X00100010

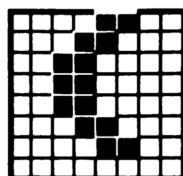

35
&H23
&X00100011

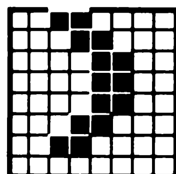

36
&H24
&X00100100

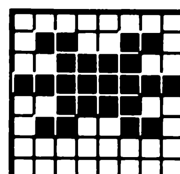

37
&H25
&X00100101

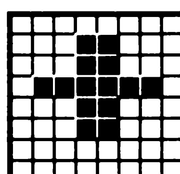

38
&H26
&X00100110

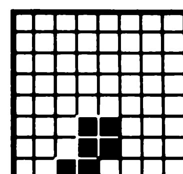

39
&H27
&X00100111

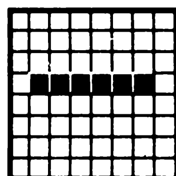

40
&H28
&X00101000

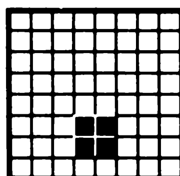

41
&H29
&X00101001

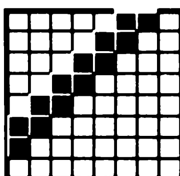

42
&H2A
&X00101010

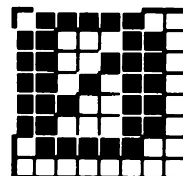

43
&H2B
&X00101011

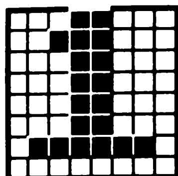

44
&H2C
&X00101100


45
&H2D
&X00101101

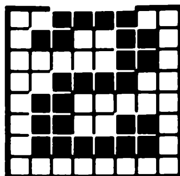

46
&H2E
&X00101110


47
&H2F
&X00101111

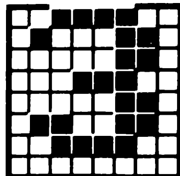

48
&H30
&X00110000



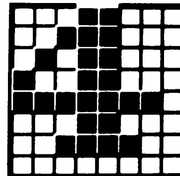
49
&H31
&X00110001



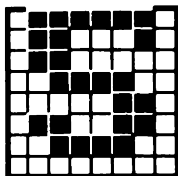
50
&H32
&X00110010



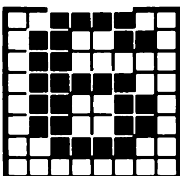
51
&H33
&X00110011



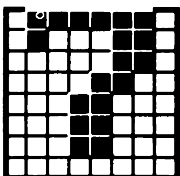
52
&H34
&X00110100



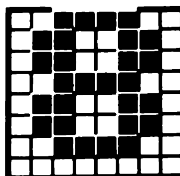
53
&H35
&X00110101



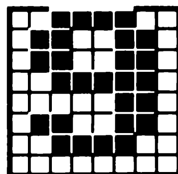
54
&H36
&X00110110



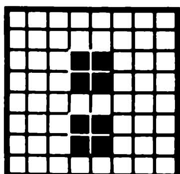
55
&H37
&X00110111



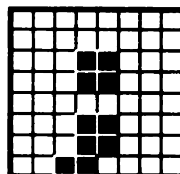
56
&H38
&X00111000



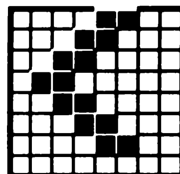
57
&H39
&X00111001



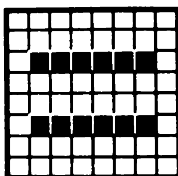
58
&H3A
&X00111010



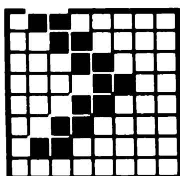
59
&H3B
&X00111011



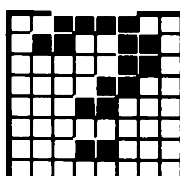
60
&H3C
&X00111100



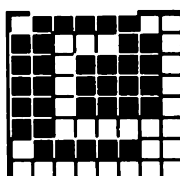
61
&H3D
&X00111101



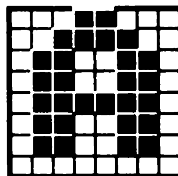
62
&H3E
&X00111110



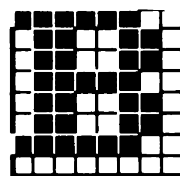
63
&H3F
&X00111111



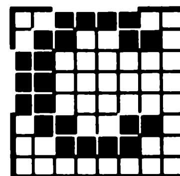
64
&H40
&X01000000



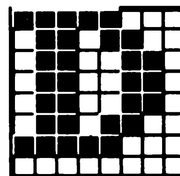
65
&H41
&X01000001



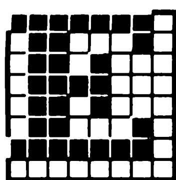
66
&H42
&X01000010

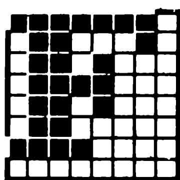


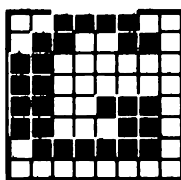
67
&H43
&X01000011

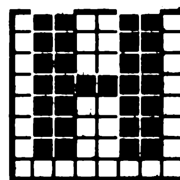


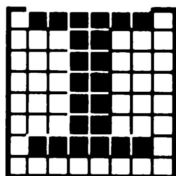
68
&H44
&X01000100

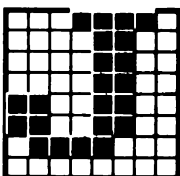

69
&H45
&X01000101

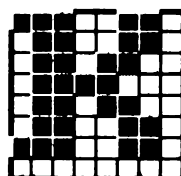

70
&H46
&X01000110

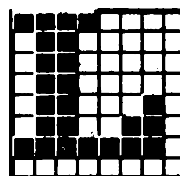

71
&H47
&X01000111

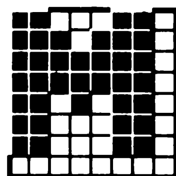

72
&H48
&X01001000

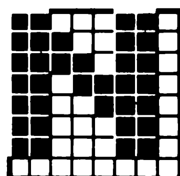

73
&H49
&X01001001

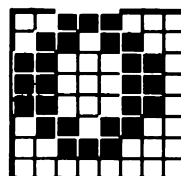

74
&H4A
&X01001010

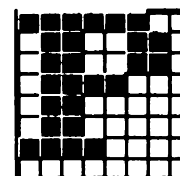

75
&H4B
&X01001011

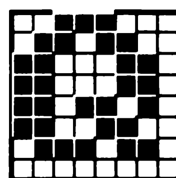

76
&H4C
&X01001100

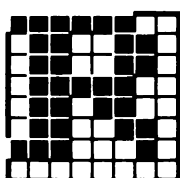

77
&H4D
&X01001101

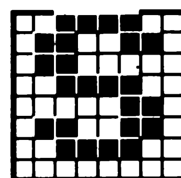

78
&H4E
&X01001110

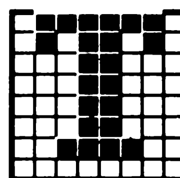

79
&H4F
&X01001111

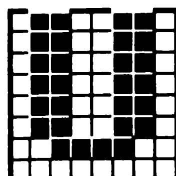

80
&H50
&X01010000

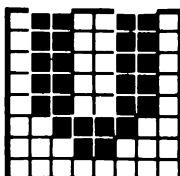

81
&H51
&X01010001

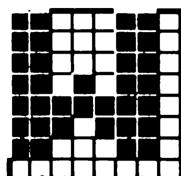

82
&H52
&X01010010

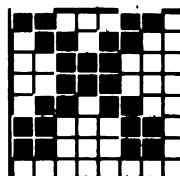

83
&H53
&X01010011

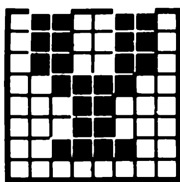

84
&H54
&X01010100

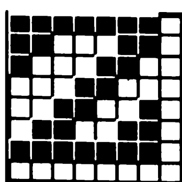

85
&H55
&X01010101

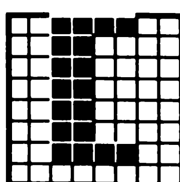

86
&H56
&X01010110

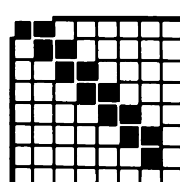

87
&H57
&X01010111

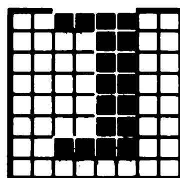

88
&H58
&X01011000

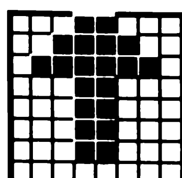

89
&H59
&X01011001

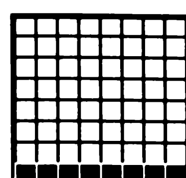

90
&H5A
&X01011010

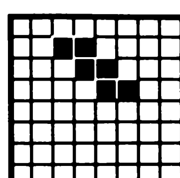

91
&H5B
&X01011011

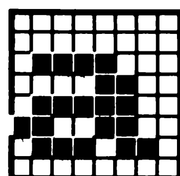

92
&H5C
&X01011100

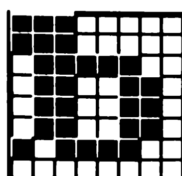

93
&H5D
&X01011101

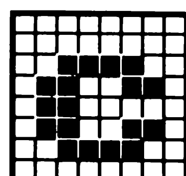

94
&H5E
&X01011110

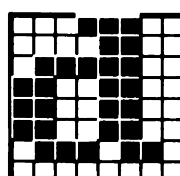

95
&H5F
&X01011111

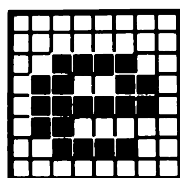

96
&H60
&X01100000

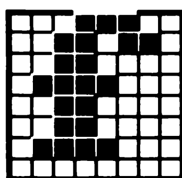

97
&H61
&X01100001

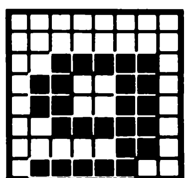

98
&H62
&X01100010

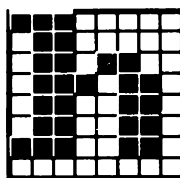

99
&H63
&X01100011

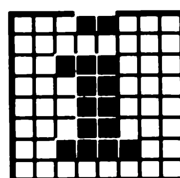

100
&H64
&X01100100

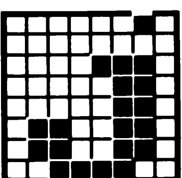

101
&H65
&X01100101

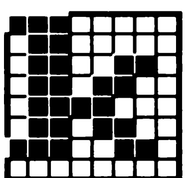

102
&H66
&X01100110

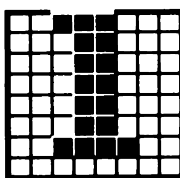

103
&H67
&X01100111

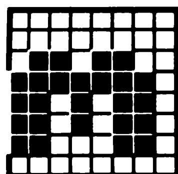

104
&H68
&X01101000

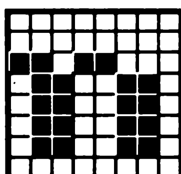

105
&H69
&X01101001

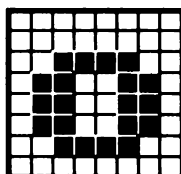

106
&H6A
&X01101010

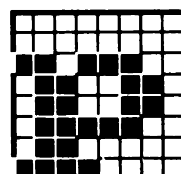

107
&H6B
&X01101011

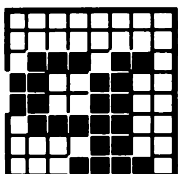

108
&H6C
&X01101100

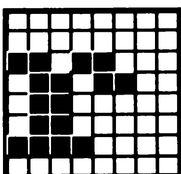

109
&H6D
&X01101101

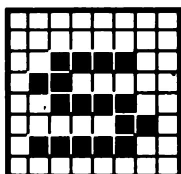

110
&H6E
&X01101110

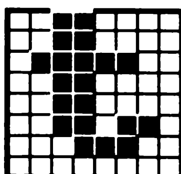

111
&H6F
&X01101111

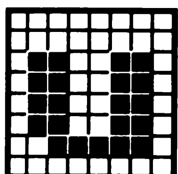

112
&H70
&X01110000

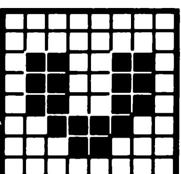

113
&H71
&X01110001

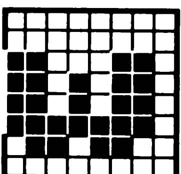

114
&H72
&X01110010

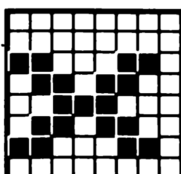

115
&H73
&X01110011

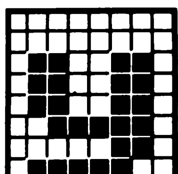

116
&H74
&X01110100

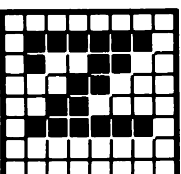

117
&H75
&X01110101

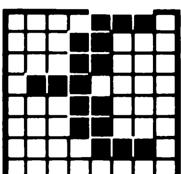

118
&H76
&X01110110

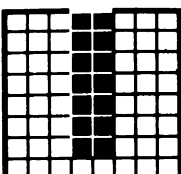

119
&H77
&X01110111

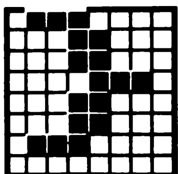

120
&H78
&X01111000

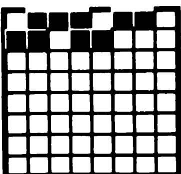

121
&H79
&X01111001

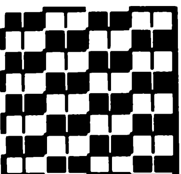

122
&H7A
&X01111010

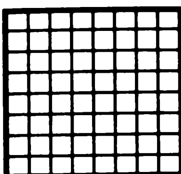

123
&H7B
&X01111011

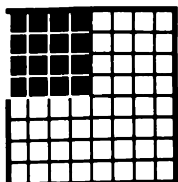

124
&H7C
&X01111100

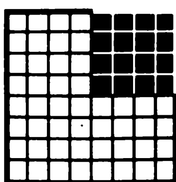

125
&H7D
&X01111101

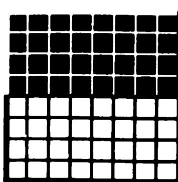

126
&H7E
&X01111110

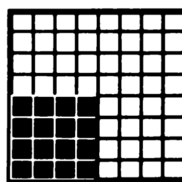

127
&H7F
&X01111111

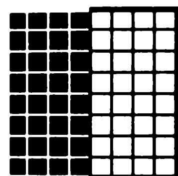

128
&H80
&X10000000

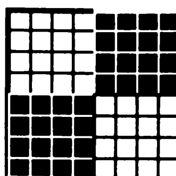

129
&H81
&X10000001

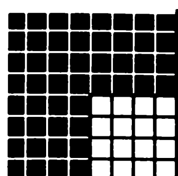

130
&H82
&X10000010

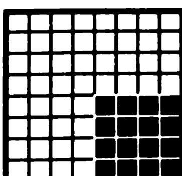

131
&H83
&X10000011

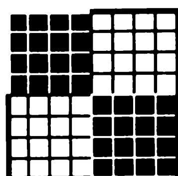

132
&H84
&X10000100

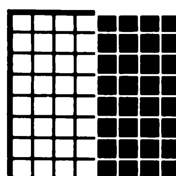

133
&H85
&X10000101

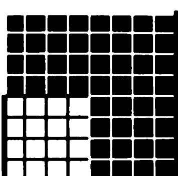

134
&H86
&X10000110

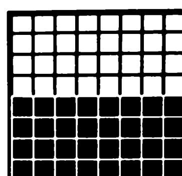

135
&H87
&X10000111

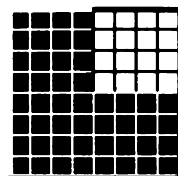

136
&H88
&X10001000

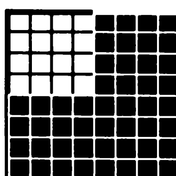

137
&H89
&X10001001

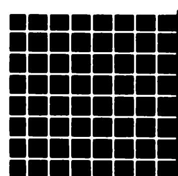

138
&H8A
&X10001010

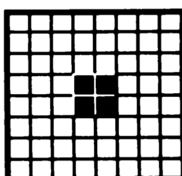

139
&H8B
&X10001011

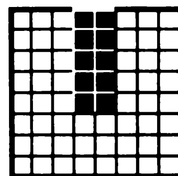

140
&H8C
&X10001100

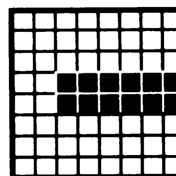

141
&H8D
&X10001101

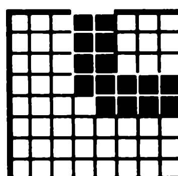

142
&H8E
&X10001110

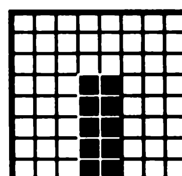

143
&H8F
&X10001111

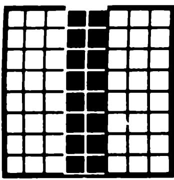

144
&H90
&X10010000

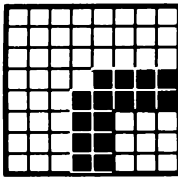

145
&H91
&X10010001

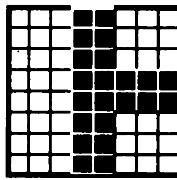

146
&H92
&X10010010

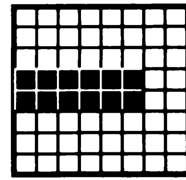

147
&H93
&X10010011

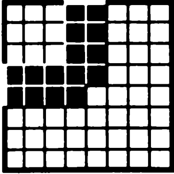

148
&H94
&X10010100

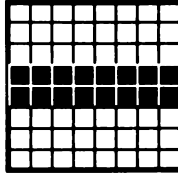

149
&H95
&X10010101

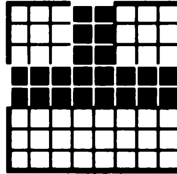

150
&H96
&X10010110

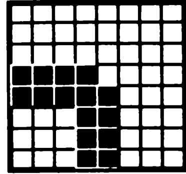

151
&H97
&X10010111

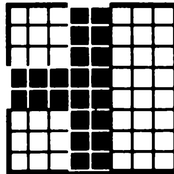

152
&H98
&X10011000

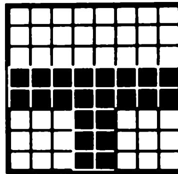

153
&H99
&X10011001

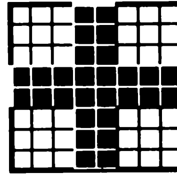

154
&H9A
&X10011010

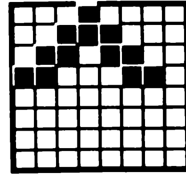

155
&H9B
&X10011011

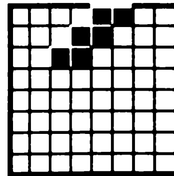

156
&H9C
&X10011100

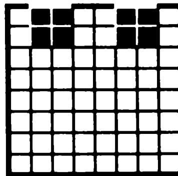

157
&H9D
&X10011101

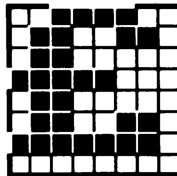

158
&H9E
&X10011110

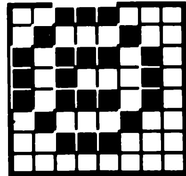

159
&H9F
&X10011111

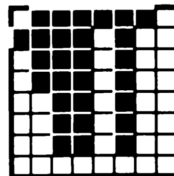

160
&HA0
&X10100000

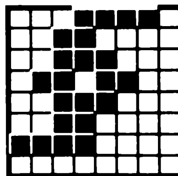

161
&HA1
&X10100001

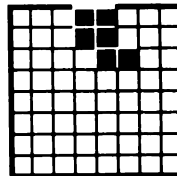

162
&HA2
&X10100010

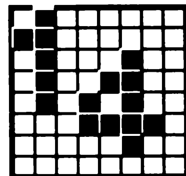

163
&HA3
&X10100011

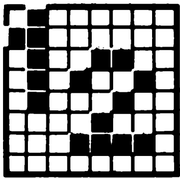

164
&HA4
&X10100100

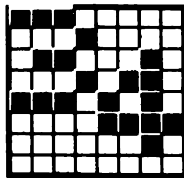

165
&HA5
&X10100101

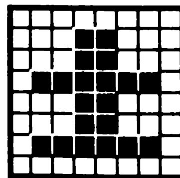

166
&HA6
&X10100110

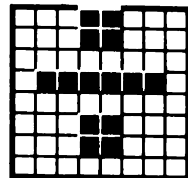

167
&HA7
&X10100111

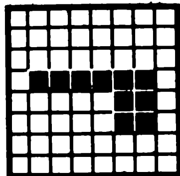

168
&HA8
&X10101000

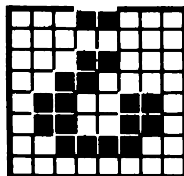

169
&HA9
&X10101001

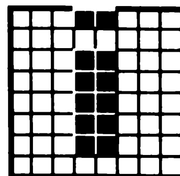

170
&HAA
&X10101010

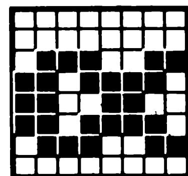

171
&HAB
&X10101011

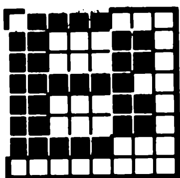

172
&HAC
&X10101100

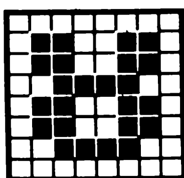

173
&HAD
&X10101101

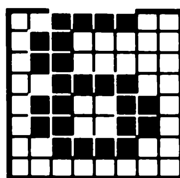

174
&HAE
&X10101110

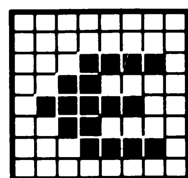

175
&HAF
&X10101111

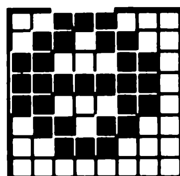

176
&HB0
&X10110000

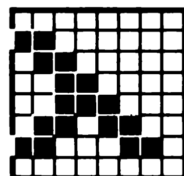

177
&HB1
&X10110001

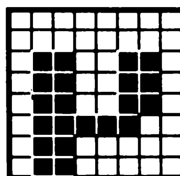

178
&HB2
&X10110010

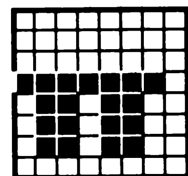

179
&HB3
&X10110011

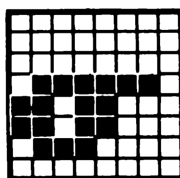

180
&HB4
&X10110100

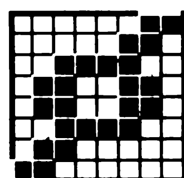

181
&HB5
&X10110101

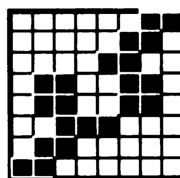

182
&HB6
&X10110110

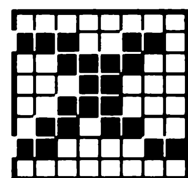

183
&HB7
&X10110111

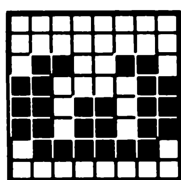

184
&HB8
&X10111000

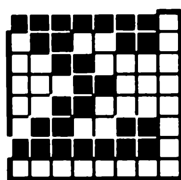

185
&HB9
&X10111001

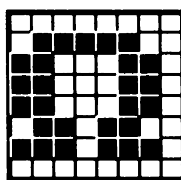

186
&HBA
&X10111010

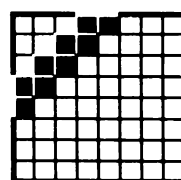

187
&HBB
&X10111011

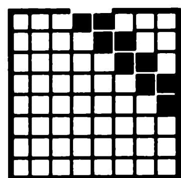

188
&HBC
&X10111100

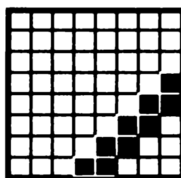

189
&HBD
&X10111101

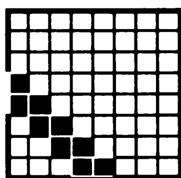

190
&HBE
&X10111110

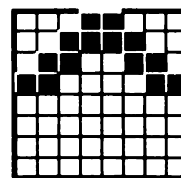

191
&HBF
&X10111111

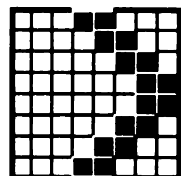

192
&HC0
&X11000000

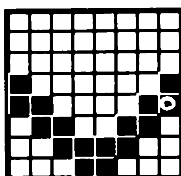

193
&HC1
&X11000001

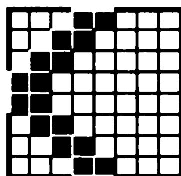

194
&HC2
&X11000010

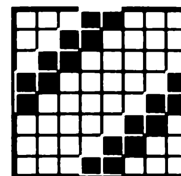

195
&HC3
&X11000011

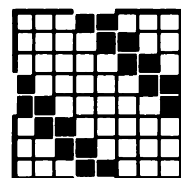

196
&HC4
&X11000100

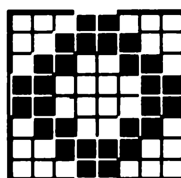

197
&HC5
&X11000101

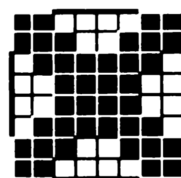

198
&HC6
&X11000110

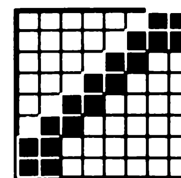

199
&HC7
&X11000111

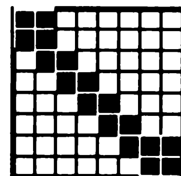

200
&HC8
&X11001000

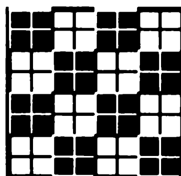

201
&HC9
&X11001001

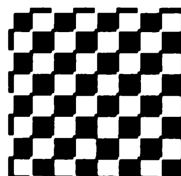

202
&HCA
&X11001010

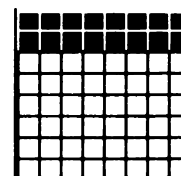

203
&HCB
&X11001011

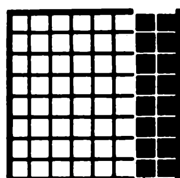

204
&HCC
&X11001100

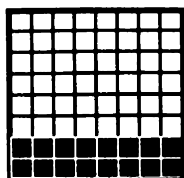

205
&HCD
&X11001101

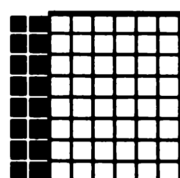

206
&HCE
&X11001110

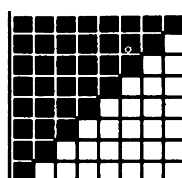

207
&HCF
&X11001111

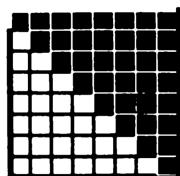

208
&HD0
&X11010000

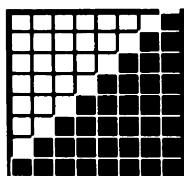

209
&HD1
&X11010001

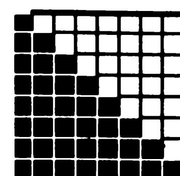

210
&HD2
&X11010010

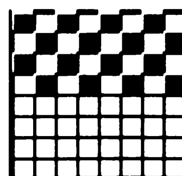

211
&HD3
&X11010011

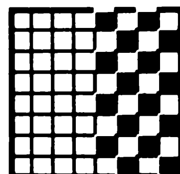

212
&HD4
&X11010100

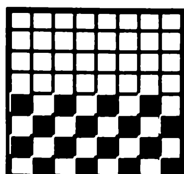

213
&HD5
&X11010101

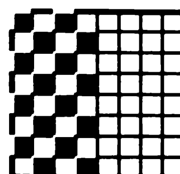

214
&HD6
&X11010110

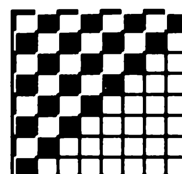

215
&HD7
&X11010111

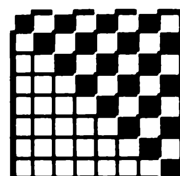

216
&HD8
&X11011000

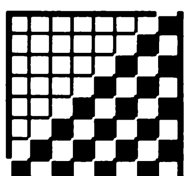

217
&HD9
&X11011001

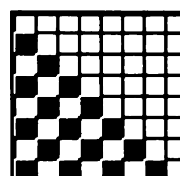

218
&HDA
&X11011010

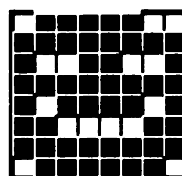

219
&HDB
&X11011011

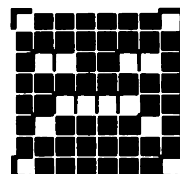

220
&HDC
&X11011100

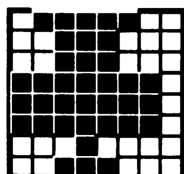

221
&HDD
&X11011101

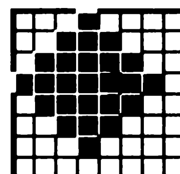

222
&HDE
&X11011110

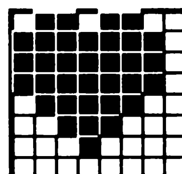

223
&HDF
&X11011111

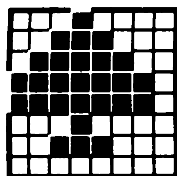

224
&HE0
&X11100000

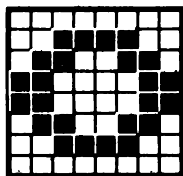

225
&HE1
&X11100001

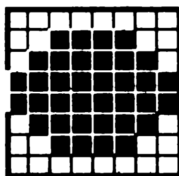

226
&HE2
&X11100010

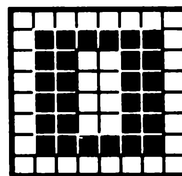

227
&HE3
&X11100011

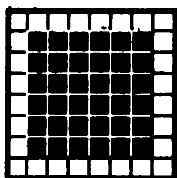

228
&HE4
&X11100100

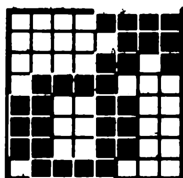

229
&HE5
&X11100101

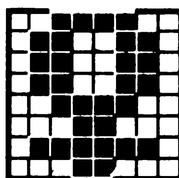

230
&HE6
&X11100110

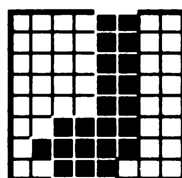

231
&HE7
&X11100111

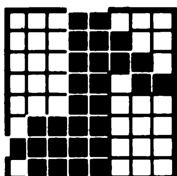

232
&HE8
&X11101000

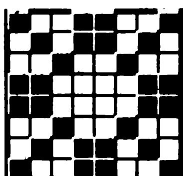

233
&HE9
&X11101001

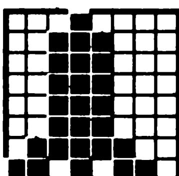

234
&HEA
&X11101010

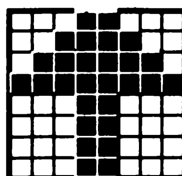

235
&HEB
&X11101011

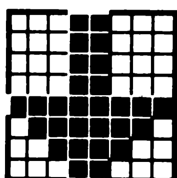

236
&HEC
&X11101100

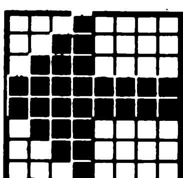

237
&HED
&X11101101

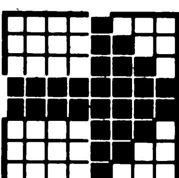

238
&HEE
&X11101110

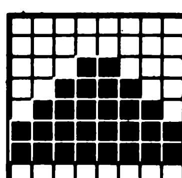

239
&HEF
&X11101111

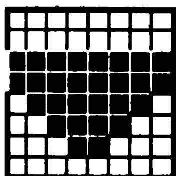

240
&HF0
&X11110000

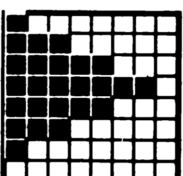

241
&HF1
&X11110001

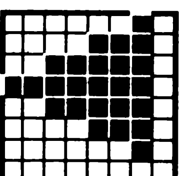

242
&HF2
&X11110010

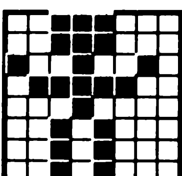

243
&HF3
&X11110011

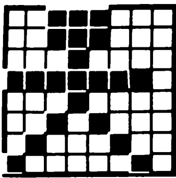

244
&HF4
&X11110100


245
&HF5
&X11110101

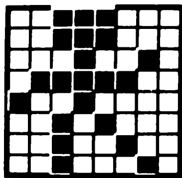

246
&HF6
&X11110110


247
&HF7
&X11110111

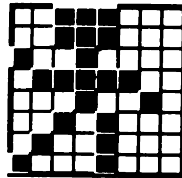

248
&HF8
&X11111000



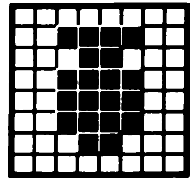
249
&HF9
&X11111001



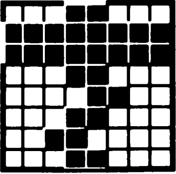
250
&HFA
&X11111010



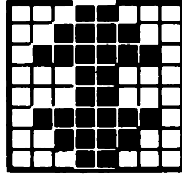
251
&HFB
&X11111011



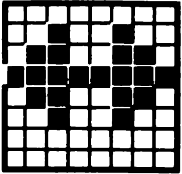
252
&HFC
&X11111100



253
&HFD
&X11111101

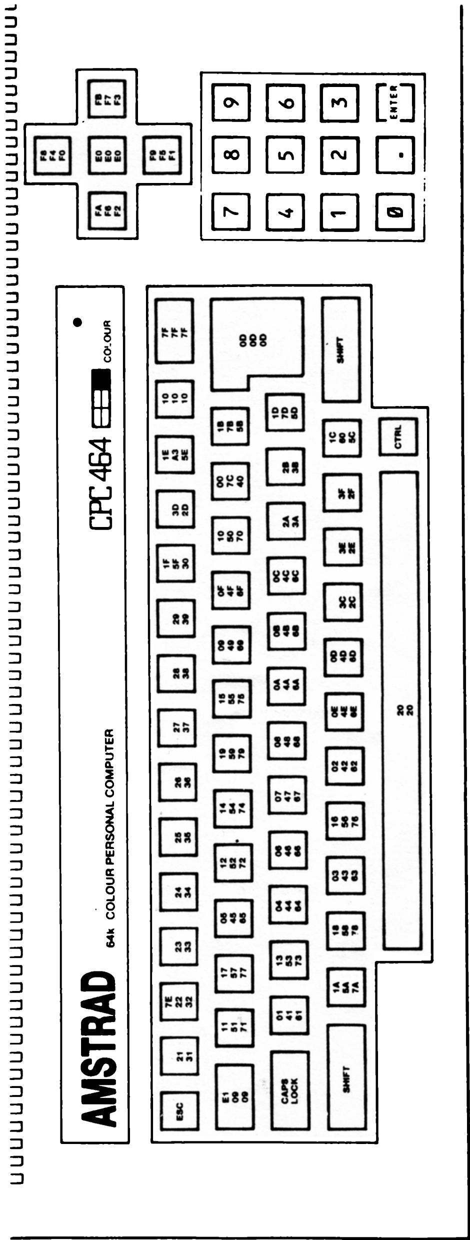


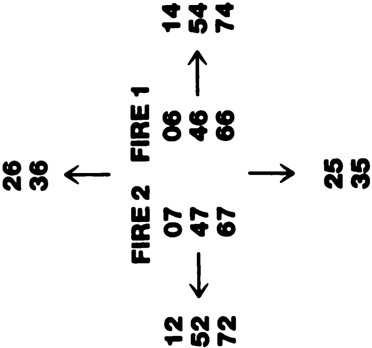
254
&HFE
&X11111110

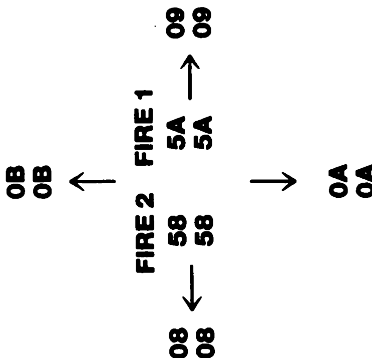


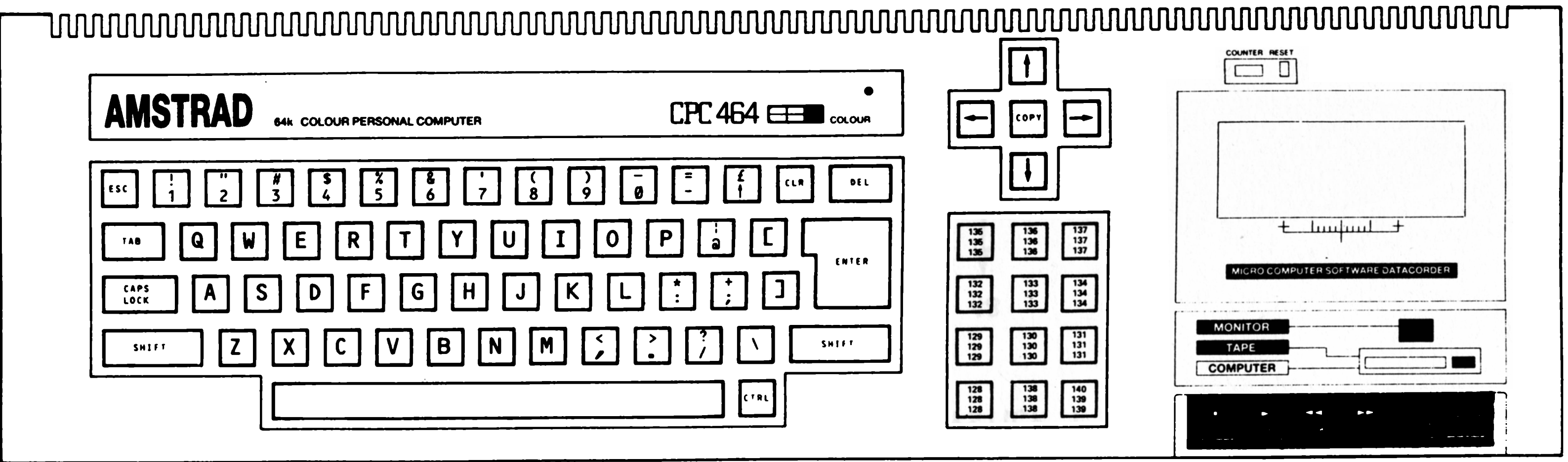
255
&HFF
&X11111111

Normali valori ASCII (esadecimali)





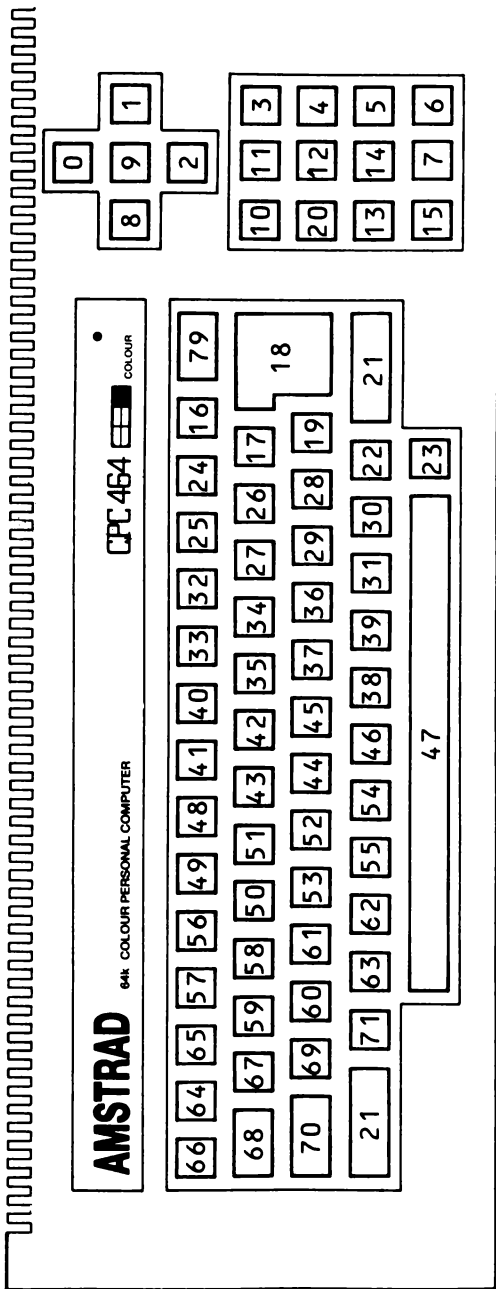


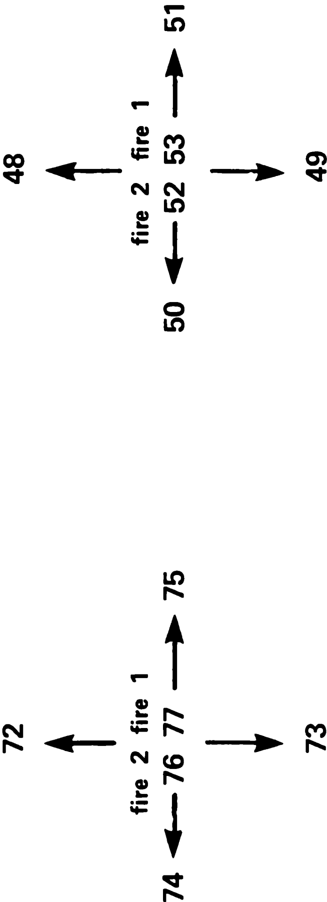


car esp	valore	ascii
128	0	30
129	1	31
130	2	32
131	3	33
132	4	34
133	5	35
134	6	36
135	7	37
136	8	38
137	9	39
138	.	2E
139	[enter]	0D
140	run"	52,55,4E,22,0D

Caratteri di espansione, normali locazioni e valori

Tasti e numeri dei joystick





Appendice IV:

Introduzione all'uso del CPC464 per utilizzatori esperti

Il CPC464 è un personal computer a colori di basso costo che contiene una vasta, potente implementazione delle tecnologie acquisite, presentato in un formato progettato per offrire uno straordinario rapporto qualità prezzo e notevoli capacità di espansione in modo da soddisfare sia il principiante che l'esperto.

L'hardware ed il software su ROM sono stati progettati per fornire sia ai principianti che agli esperti programmatori un ambiente di uso semplice in cui il software esistente può essere personalizzato e può essere scritto del nuovo software per utilizzare a fondo le caratteristiche fornite dal CPC464.

Le caratteristiche principali del sistema sono:

CPU Z80

Il microprocessore più usato in tutto il mondo per i computer domestici, con la base software più sviluppata (il CPC464 permette di implementare il CP/M). Una struttura di interrupt unica nel suo genere estende il BASIC mediante le parole chiave AFTER ed EVERY oltre ad altre caratteristiche "real-time" che controllano il suono ed i timer.

64K di RAM

Viene fornito uno spazio di RAM di 64K, dei quali più di 42K sono effettivamente utilizzabili dall'utente, grazie a tecniche di overlay utilizzate nell'implementazione del BASIC.

Schermo

Il CPC464 permette di utilizzare tre modalità, che permettono di avere uno schermo ad 80 colonne di testo, una tavolozza di 27 colori ed una risoluzione di 640x200 pixel.

Una vera tastiera

Una completa tastiera tipo macchina per scrivere, con tasti per il controllo del cursore, un tastierino numerico con possibilità di usare i tasti funzione.

Cassetta incorporata

E' possibile memorizzare su cassetta il proprio lavoro utilizzando il registratore per dati incorporato. In tal modo è possibile fare qualunque operazione senza preoccuparsi del livello audio, delle connessioni, ecc. E' possibile scrivere dati ad una velocità di 1 o 2k baud (selezionabile via software) mentre la velocità di lettura è stabilita automaticamente dal software.

BASIC

Un BASIC standard, più veloce e versatile di quanto ci si potrebbe mai aspettare, con molte estensioni per la grafica ed il suono, e possibilità di accesso al firmware.

Set di caratteri esteso

Un set di caratteri ad 8-bit che comprende simboli ed elementi grafici accessibili in larga misura da tastiera ed utilizzando la funzione CHR$(n).

Tempo

Gli interrupt sono generati dalle scansioni dei frame e permettono il controllo del tempo trascorso.

Tasti definibili dall'utente

Possono essere definiti dall'utente fino a 32 tasti; ognuno di essi può produrre una stringa di 32 caratteri. Le possibilità di ridefinizione includono i parametri di ripetizione. E' inoltre presente un completo set di 255 caratteri (tutti gli ASCII più altri 100), che può essere ridefinito dall'utente.

Subroutine

E' possibile richiamare da BASIC molte sub-routine scritte in assembler.

MODALITA' DI SCHERMO

Vi sono tre diverse modalità di schermo:

a) Normale
Modalità 1: 40 colonnex25 linee, 4 colori per il testo, 320x200 pixel indirizzabili individualmente e 4 colori.

b) Colori multipli
20 colonnex25 linee, 16 colori per il testo, 160x200 pixel indirizzabili individualmente e 16 colori.

c) Alta risoluzione
80 colonnex25 linee, 2 colori per il testo, 640x200 pixel indirizzabili individualmente e 2 colori.

Scelta dei colori

NB All'interno della guida, il nero, ovvero lo schermo senza alcuna luminosità, è considerato come un colore.

La cornice dello schermo può assumere una qualunque coppia di colori, indipendentemente dalla modalità dello schermo.

Il numero dei colori disponibili sullo schermo dipende invece dalla modalità scelta. Il colore può essere scelto in coppia con un altro per ottenere un lampeggiamento o può essere fisso. I comandi INK, PAPER per il testo, PEN per il testo e PEN per la grafica permettono di scegliere uno qualunque dei colori disponibili.

La scrittura del testo può essere traslucida o opaca: in un caso verrà ignorato il colore dello sfondo e la grafica verrà sovrascritta, nell'altro caso lo sfondo verrà completamente sovrascritto.

Finestre

L'utente può selezionare fino ad 8 finestre di testo in cui scrivere i caratteri ed 1 finestra grafica per i disegni.

Quando si cambia modalità di schermo, le finestre vengono riportate al loro stato iniziale.

NB: Se la finestra di testo corrisponde all'intero schermo (come la condizione normale), l'hardware permette uno scorrimento rapido dei caratteri. Se la finestra di testo è più piccola dello schermo disponibile, lo scorrimento viene gestito dal software che è leggermente più lento.

Cursore

Il cursore viene disabilitato se la CPU non attende alcun input da tastiera, ed agisce perciò come un prompt automatico. Il cursore è costituito da un rettangolo a colori invertiti.

Suono polifonico

Le possibilità sonore del CPC464 sono permesse dall'uso del generatore sonoro che appartiene alla famiglia AY8910 della General Instrument. Questo dispositivo opera su tre canali (voci) per ognuna delle quali è possibile regolare il tono ed il volume. Se occorre può essere aggiunto del rumore bianco.

I tre canali appaiono a sinistra, a destra ed al centro (usando la presa jack). L'altoparlante interno fornisce un suono miscelato mono.

Il software permette la gestione e il controllo degli inviluppi per gestire toni e volume. Il sistema di inviluppi del generatore sonoro non viene invece normalmente usato.

Porta per stampante

Viene fornita una porta per stampanti parallele standard, che utilizza il segnale "Busy" per gestire le operazioni di handshake.

Possibilità di espansione

Sono disponibili molte interfacce hardware e vi è la possibilità di inserire espansioni software. La AMSTRAD intende presentare, tra l'altro, drive per dischi, interfacce seriali con software di gestione in ROM, ecc.

Coordinate

L'origine per il testo è l'angolo in alto a sinistra dello schermo e le posizioni fisiche dello schermo cambiano a seconda della modalità utilizzata.

L'origine grafica è invece posta nell'angolo in basso a sinistra ed assumerà che lo schermo sia sempre in alta risoluzione - ma i calcoli relativi ai colori verranno effettuati correttamente a seconda della modalità impostata.

NB

Con lo schermo nella modalità normale, ogni pixel ha due indirizzi orizzontali e può essere usato uno qualunque dei due. In modalità a colori multipli, ogni pixel ha quattro indirizzi orizzontali e, per definire la posizione del pixel può essere usato uno qualunque dei quattro.

L'asse verticale ha coordinate che vanno da 0 a 399, divise per due e troncate per fornire una posizione fisica tra 0 e 199. In questo modo vengono mantenuti i normali rapporti tra le coordinate.

ROM di espansione

Tutte le ROM occupano gli ultimi 16K della memoria (dove risiede il BASIC) ed il firmware permette di utilizzare fino ad altre 240 ROM addizionali da 16K (anche se in questo caso è necessario utilizzare una certa quantità di hardware esterno per la gestione degli indirizzi).

Panoramica:

Un breve sommario delle caratteristiche principali dell'-hardware e del firmware del CPC464.

1)Hardware

1.1 Cosa si trova dentro la scatola.
Computer, tastiera, registratore dati a cassette e altoparlante, uscite RGB e video.

1.1.1) Chip LSI (Integrazione a larga scala)
Processore Z80A con clock a 4MHz
64 Kbytes di RAM (64K x 1) con refresh ad ogni accesso alla memoria di schermo.
32 Kbytes di ROM contenenti il BASIC ed il sistema operativo.

Un chip custom contiene quasi tutta la logica di funzionamento che non si trova nei chip LSI; in particolare gli aspetti della sincronizzazione, della generazione del colore ed il circuito DMA.
Il controller video 6845 genera i segnali di scansione della RAM video.

NB
La mappa di memoria è complessa e cambia a seconda della modalità di schermo. Il controller video può essere utilizzato per eseguire scorrimenti orizzontali (di 1/40 di schermo) o verticali (verso l'alto o verso il basso di 8 linee di scansione), se viene richiesto dal software. I parametri del controller video selezionano il numero di linee, la frequenza di frame, l'ampiezza e la posizione della cornice.

Generatore sonoro General Instruments AY-3-8912: 3 voci. Il suono viene prelevato dai tre output e viene miscelato per ottenere l'audio mono fornito dall'altoparlante interno, regolato dal controllo di volume. L'uscita audio stereo è composta come segue.

Sinistro = Canale A + 1/2 del Canale C.
Destro = Canale B + 1/2 del Canale C.

Questo chip riceve anche le informazioni relative alla scansione della tastiera e dei joystick.

Un dispositivo per I/O parallelo 8255 è interfacciato con il bus del generatore sonoro. Anch'esso effettua la scansione della tastiera, dei joystick, dei dispositivi opzionali e controlla la cassetta.

1.1.2) Prese esterne.
Connettori a pettine per espansioni varie (12) e per una stampante (12) parallela Centronics. Prese per joystick (14) (tipo D a 9 pin), uscita stereo (15), uscita video (10) (RGB e sync o video composito o luminanza e sync).

1.2) All'esterno della scatola.
Il CPC464 permette di utilizzare due tipi di monitor. Entrambi possiedono una uscita di alimentazione a 5V per il computer, e sono adatti all'uso nel paese dove viene effettuato l'acquisto. Inoltre è disponibile un alimentatore-modulatore TV UHF, MP2.

Per collegare la stampante parallela ed una unità ad alta fedeltà è necessario utilizzare gli appositi cavi.

1.3) Caratteristiche dello schermo
Lo schermo utilizza 16K di memoria. La macchina può usare 27 colori selezionabili liberamente dalla tavolozza. Il numero di colori della tavolozza dipende dalla modalità di schermo impostata. Se necessario è possibile fissare più INK allo stesso colore. I pixel sullo schermo sono definiti come punti di un determinato colore. E' da notare che lo sfondo deve essere di un colore della tavolozza. Intorno alla parte attiva dello schermo vi è una cornice che è possibile colorare utilizzando uno qualunque dei 27 colori.

Modalità	Colori	Punti vert	Punti orizz	Caratteri orizz.
Normale	4	200	320	40
Alta risoluzione	2	200	640	80
Colori multipli	16	200	160	20

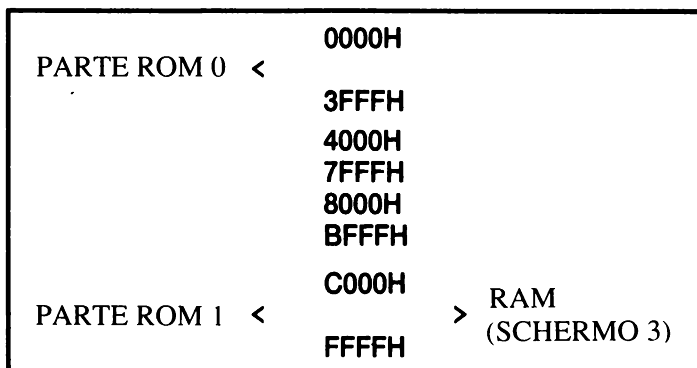
NB: Su un monitor monocromatico il computer mostrerà una scala di grigi. La luminosità dei colori è in quest'ordine:

Liv Grigio	Colore	Liv Grigio	Colore
0	Nero	13	Bianco
1	Blu	14	Blu pastello
2	Blu brillante	15	Arancio
3	Rosso	16	Rosa
4	Magenta	17	Magenta pastello
5	Malva	18	Verde brillante
6	Rosso brillante	19	Verde mare
7	Porpora	20	Azzurro brillante
8	Magenta brillante	21	Verde limone
9	Verde	22	Verde pastello
10	Azzurro	23	Azzurro pastello
11	Azzurro cielo	24	Giallo brillante
12	Giallo	25	Giallo pastello
		26	Bianco brillante

1.4) Mappa di memoria

I 64K di RAM sono allocati in questo modo.

Si noti che parte della ROM si sovrappone alla RAM video in modo da lasciare all'utente la maggior quantità possibile di RAM per l'utilizzo del BASIC.



Quando ad un indirizzo vi è sia la ROM che la RAM, la lettura viene effettuata dalla ROM e la scrittura sulla RAM. E' possibile disabilitare entrambe le parti della ROM, in modo da permettere la lettura della RAM nello stesso indirizzo.

1.5) Possibilità di espansione

1.5.1) ROM

E' previsto l'utilizzo di altre ROM al posto della ROM contenuta nella macchina. La scelta degli indirizzi e la logica di selezione dei banchi sarà contenuta in un modulo connesso al bus di espansione, ma tutti i segnali occorrenti vengono inviati al bus di espansione.

1.5.2) RAM addizionale

E' previsto l'utilizzo di altre RAM al posto di una qualunque parte della RAM interna. La scelta degli indirizzi e la logica di selezione dei banchi sarà contenuta in un modulo connesso al bus di espansione, ma tutti i segnali occorrenti vengono inviati al bus di espansione. Questa memoria sarà a sola lettura e per scrivere su questa RAM sarà necessario uno schema speciale riguardante il mappaggio dell'I/O.

1.5.3) I/O addizionale

La maggior parte degli indirizzi delle porte di I/O è riservata dal computer; in particolare non dovrebbero essere usati gli indirizzi sotto 7Fxx. I seguenti indirizzi possono invece essere utilizzati dall'hardware esterno:

F8xx, F9xx, FAxx, FBxx

Le periferiche connesse al bus di espansione devono decodificare gli indirizzi in A0 .. A7 mentre l'indirizzo A10 è basso. I canali di I/O del bus di espansione con indirizzi che vanno da F800 a FBFF sono riservati nel modo seguente:

Indirizzi A0-A7

00 - 7B	** Non usare **
7C - 7F	Riservati all'interfaccia dischi
80 - BB	** Non usare **
BC - BF	Riservati per usi futuri
C0 - DB	** Non usare **
DC - DF	Riservati per le interfacce di comunicazione
E0 - FF	Disponibili per le periferiche utente

Si noti che devono essere utilizzate le istruzioni dello Z80 che inviano il registro **B** nella parte superiore del bus indirizzi (**A15-A8**).

2) Tastiera.

Per provocare una reinizializzazione occorre premere insieme i tasti **[CTRL][SHIFT][ESC]**. I tasti che provocano la stampa di un carattere o il movimento del cursore, tranne i tasti del tastierino numerico, sono dotati di ripetizione automatica controllata dal firmware.

[ESC] sospende l'esecuzione del programma. Seguito da un altro **[ESC]** ferma definitivamente l'esecuzione. Seguito da un altro carattere riprende l'esecuzione.

[CAPS LOCK] fissa l'uso dei caratteri maiuscoli.

[CTRL][CAPS LOCK] fissa l'uso dei caratteri sulla parte superiore del tasto.

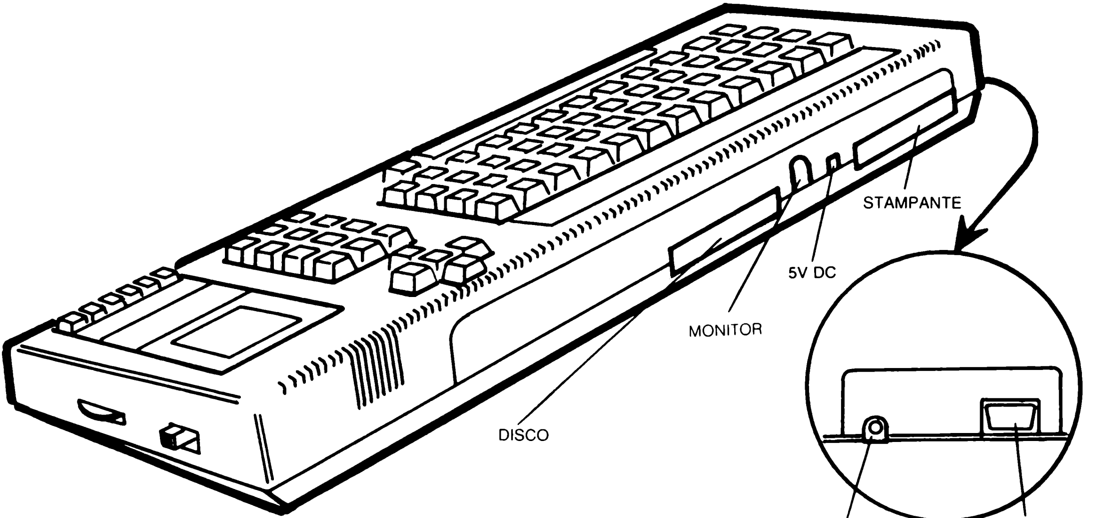
Il secondo cursore viene separato dal cursore principale premendo **[SHIFT]** insieme ad un tasto cursore. L'input verrà riportato sul carattere che segue il cursore premendo il tasto **[COPY]**.

I tasti cursore permettono la correzione di ciò che si trova nel buffer di input, che puo trovarsi disposto su più linee dello schermo. I tasti cursore possono essere utilizzati per posizionare l'inizio dell'input da tastiera in qualunque parte dello schermo, prima che venga ricevuto un qualunque input dalla tastiera. Una volta che l'input da tastiera è stato ricevuto, la posizione sullo schermo rimane fissata. L'input di un nuovo testo andrà a sovrapporsi a qualunque cosa sia contenuta in quella posizione sullo schermo.

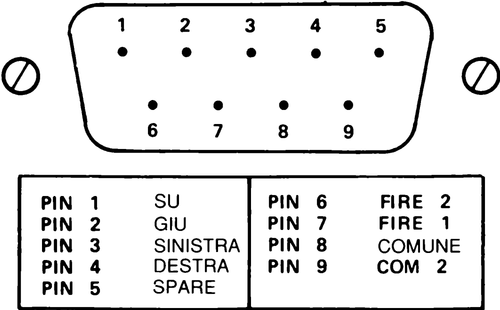
[DEL] cancella all'indietro e **[CLR]** cancella in avanti.

Appendice V:

Pannello posteriore del CPC464

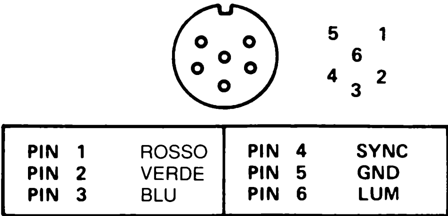


PORTA DI CONNESSIONE JOYSTICK (9 PIN D)
VISTO DAL RETRO



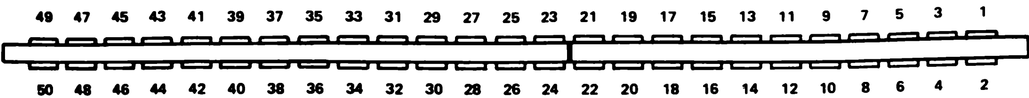
PIN 1	SU	**PIN 6**	**FIRE 2**
PIN 2	GIU	**PIN 7**	**FIRE 1**
PIN 3	SINISTRA	**PIN 8**	COMUNE
PIN 4	DESTRA	**PIN 9**	**COM 2**
PIN 5	SPARE		

CONNET E DI OUTPUT VIDEO (6 PIN DIN)
VISTA DAL RETRO



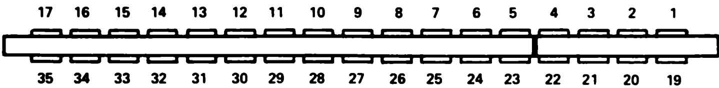
PIN 1	ROSSO	**PIN 4**	**SYNC**
PIN 2	VERDE	**PIN 5**	**GND**
PIN 3	BLU	**PIN 6**	**LUM**

PORTA DI ESPANSIONE CONNETTORE A 50 POLI
VISTO DAL RETRO



PIN 1	SOUND	PIN 18	A0	PIN 35	INT
PIN 2	GND	PIN 19	D7	PIN 36	NMI
PIN 3	A15	PIN 20	D6	PIN 37	BUSRD
PIN 4	A14	PIN 21	D5	PIN 38	BUSAK
PIN 5	A13	PIN 22	D4	PIN 39	READY
PIN 6	A12	PIN 23	D3	PIN 40	BUS RESET
PIN 7	A11	PIN 24	D2	PIN 41	RESET
PIN 8	A10	PIN 25	D1	PIN 42	ROMEN
PIN 9	A9	PIN 26	D0	PIN 43	ROMDIS
PIN 10	A8	PIN 27	+ 5v	PIN 44	RAMRD
PIN 11	A7	PIN 28	MREQ	PIN 45	RAMDIS
PIN 12	A6	PIN 29	M1	PIN 46	CURSOR
PIN 13	A5	PIN 30	RFSH	PIN 47	L. PEN
PIN 14	A4	PIN 31	IORQ	PIN 48	EXP
PIN 15	A3	PIN 32	RD	PIN 49	GND
PIN 16	A2	PIN 33	WR	PIN 50	ϕ
PIN 17	A1	PIN 34	HALT		

PORTA PER STAMPANTE CONNETTORE A 34 POLI
VISTO DAL RETRO



PIN 1	STROBE	PIN 19	GND
PIN 2	D0	PIN 20	GND
PIN 3	D1	PIN 21	GND
PIN 4	D2	PIN 22	GND
PIN 5	D3	PIN 23	GND
PIN 6	D4	PIN 24	GND
PIN 7	D5	PIN 25	GND
PIN 8	D6	PIN 26	GND
PIN 9	GND	PIN 28	GND
PIN 11	BUSY	PIN 33	GND
PIN 14	GND		
PIN 16	GND		

TUTTI GLI ALTRI POLI=NON CONNESSI

GATE ARRAY
R
V
B
SYNC
+5
680
220
1 ROSSO
+5
680
220
2 VERDE
+5
680
220
3 BLU
3k3
10k
1K
+5
100
6 LUM
220
0
680
220
4 SYNC
0
5 GND

CONNETTORE
DI OUTPUT VIDEO
(6 PIN DIN)

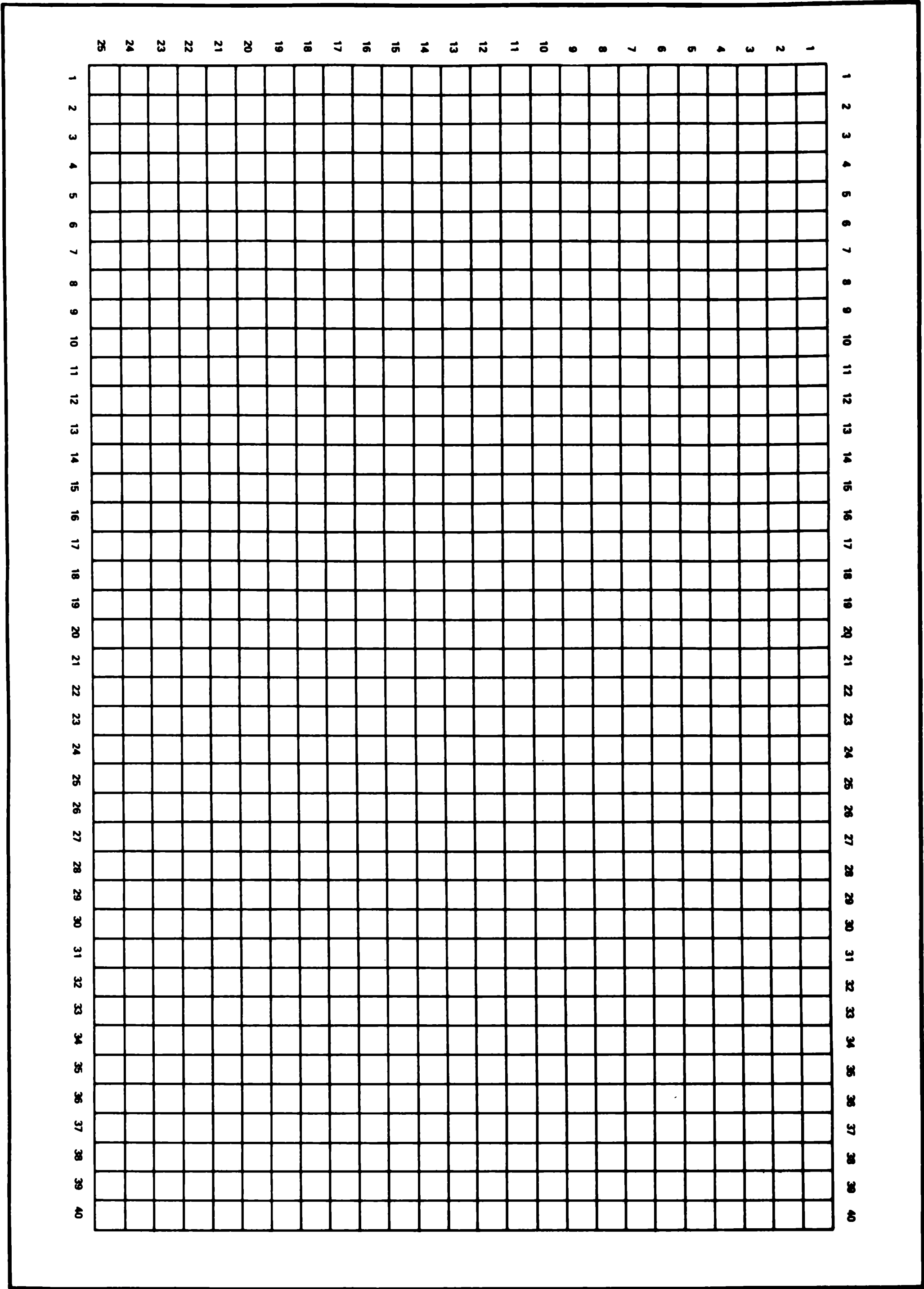
GENERATORE
SONORO
A
B
C
SUONO ALL'AMPLIFICATORE
INTERNO E ALL' ALTOPARLANTE
10K 10K 10K
A
B
C
ALL'USCITA SUONO
(ESPANSIONE)
10K 22K 10K
1K
CONNETTORE STEREO
(Jack stereo)
GND

Appendice VI:

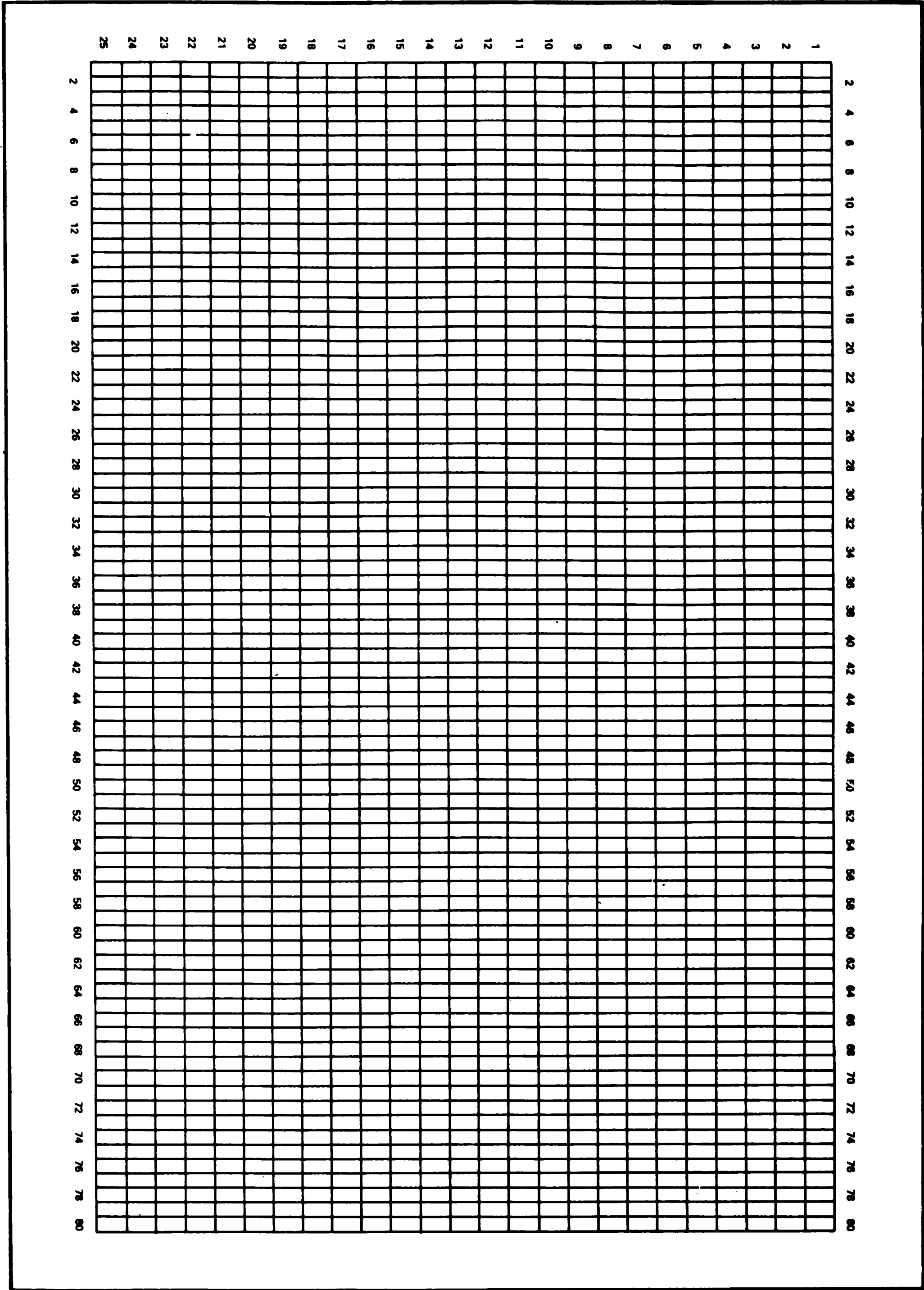
Schermo TESTO e FINESTRA
MODE 0 20 colonne

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1																				
2																				
3																				
4																				
5																				
6																				
7																				
8																				
9																				
10																				
11																				
12																				
13																				
14																				
15																				
16																				
17																				
18																				
19																				
20																				
21																				
22																				
23																				
24																				
25																				

Schermo TESTO e FINESTRA MODE 1 40 colonne



Schermo TESTO e FINESTRA
MODE 2 80 colonne



Appendice VII: Periodi di note e toni.

La tabella che segue, fornisce i periodi consigliati per le note nella normale scala temperata, per tutte le otto ottave.

La frequenza prodotta non è esattamente la frequenza richiesta in quanto il periodo deve essere composto da un intero. L'ERRORE RELATIVO è il rapporto percentuale della differenza tra la frequenza corretta e quella effettivacioè (CORRETTA-EFFETTIVA)/ CORRETTA).

NOTA	FREQUENZA	PERIODO	ERRORE RELATIVO	
Do	32.703	3822	-0.007%	
Do #	34.648	3608	+0.007%	
Re	36.708	3405	-0.007%	
Re #	38.891	3214	-0.004%	
Mi	41.203	3034	+0.009%	
Fa	43.654	2863	-0.016%	
Fa #	46.249	2703	+0.009%	Ottava 3
Sol	48.999	2551	-0.002%	
Sol #	51.913	2408	+0.005%	
La	55.000	2273	+0.012%	
La #	58.270	2145	-0.008%	
Si	61.735	2025	+0.011%	

NOTA	FREQUENZA	PERIODO	ERRORE RELATIVO	
Do	65.406	1911	-0.007%	
Do #	69.296	1804	+0.007%	
Re	73.416	1703	+0.022%	
Re #	77.782	1607	-0.004%	
Mi	82.407	1517	+0.009%	
Fa	87.307	1432	+0.019%	Ottava 2
Fa #	92.499	1351	-0.028%	
Sol	97.999	1276	+0.037%	
Sol #	103.826	1204	+0.005%	
La	110.000	1136	-0.032%	
La #	116.541	1073	+0.039%	
Si	123.471	1012	-0.038%	

NOTA	FREQUENZA	PERIODO	ERRORE RELATIVO	
Do	130.813	956	+0.046%	
Do #	138.591	902	+0.007%	
Re	146.832	851	-0.037%	
Re #	155.564	804	+0.058%	
Mi	164.814	758	-0.057%	
Fa	174.614	716	+0.019%	Ottava-1
Fa #	184.997	676	+0.046%	
Sol	195.998	638	+0.037%	
Sol #	207.652	602	+0.005%	
La	220.000	568	-0.032%	
La #	233.082	536	-0.055%	
Si	246.942	506	-0.038%	

NOTA	FREQUENZA	PERIODO	ERRORE RELATIVO	
Do	261.626	478	+0.046%	Do centrale
Do #	277.183	451	+0.007%	
Re	293.665	426	+0.081%	
Re #	311.127	402	+0.058%	
Mi	329.628	379	-0.057%	
Fa	349.228	358	+0.019%	Ottava 0
Fa #	369.994	338	+0.046%	
Sol	391.995	319	+0.037%	
Sol #	415.305	301	+0.005%	
La	440.000	284	-0.032%	
La #	466.164	268	-0.055%	La internazionale
Si	493.883	253	-0.038%	

NOTA	FREQUENZA	PERIODO	ERRORE RELATIVO	
Do	523.251	239	+0.046%	
Do #	554.365	225	-0.215%	
Re	587.330	213	+0.081%	
Re #	622.254	201	+0.058%	
Mi	659.255	190	+0.206%	
Fa	698.457	179	+0.019%	Ottava 1
Fa #	739.989	169	+0.046%	
Sol	783.991	159	-0.277%	
Sol #	830.609	150	-0.328%	
La	880.000	142	-0.032%	
La #	932.328	134	-0.055%	
Si	987.767	127	+0.356%	

NOTA	FREQUENZA	PERIODO	ERRORE RELATIVO	
Do	1046.502	119	-0.374%	
Do #	1108.731	113	+0.229%	
Re	1174.659	106	-0.390%	
Re #	1244.508	100	-0.441%	
Mi	1318.510	95	+0.206%	
Fa	1396.913	89	-0.543%	Ottava 2
Fa #	1479.978	84	-0.548%	
Sol	1567.982	80	+0.350%	
Sol #	1661.219	75	-0.328%	
La	1760.000	71	-0.032%	
La #	1864.655	67	-0.055%	
Si	1975.533	63	-0.435%	

NOTA	FREQUENZA	PERIODO	ERRORE RELATIVO	
Do	2093.004	60	+0.462%	
Do #	2217.461	56	-0.662%	
Re	2349.318	53	-0.390%	
Re #	2489.016	50	-0.441%	
Mi	2637.021	47	-0.855%	
Fa	2793.826	45	+0.574%	Ottava 3
Fa #	2959.955	42	-0.548%	
Sol	3135.963	40	+0.350%	
Sol #	3322.438	38	+0.992%	
La	3520.000	36	+1.357%	
La #	3729.310	34	+1.417%	
Si	3951.066	32	+1.134%	

NOTA	FREQUENZA	PERIODO	ERRORE RELATIVO	
Do	4186.009	30	+0.462%	
Do #	4434.922	28	-0.662%	
Re	4698.636	27	+1.469%	
Re #	4978.032	25	-0.441%	
Mi	5274.041	24	+1.246%	
Fa	5587.652	22	-1.685%	Ottava 4
Fa #	5919.911	21	-0.548%	
Sol	6271.927	20	+0.350%	
Sol #	6644.875	19	+0.992%	
La	7040.000	18	+1.357%	
La #	7458.621	17	+1.417%	
Si	7902.133	16	+1.134%	

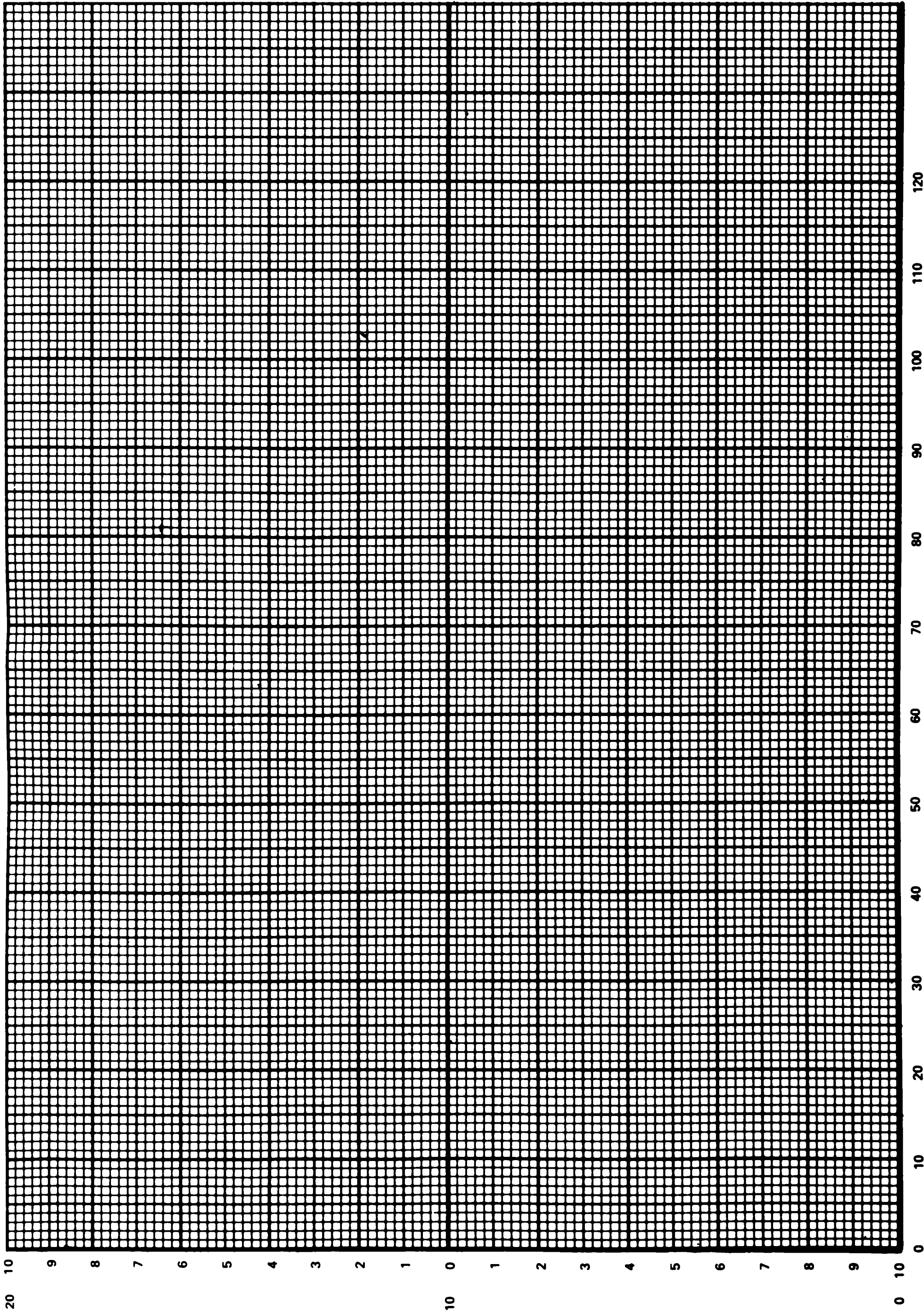
I valori sopra riportati sono tutti calcolati dal La internazionale nel modo seguente:

FREQUENZA = 440*(2^(OTTAVA+((n-10)/12)))

PERIODO = ROUND(125000/FREQUENZA)

DOVE N è 1 per il DO, 2 per il DO#, 3 per il Re, ecc.

Tabella per gli inviluppi e la musica



Appendice VIII: Codici di errore e parole riservate

Numero dell'errore e messaggi di errore

Quando il BASIC incontra un'istruzione del programma, una parola o una variabile che non può comprendere o utilizzare, si ferma e mostra un messaggio di errore. Il messaggio indica normalmente ciò che non è andato per il verso giusto - e talvolta, se l'errore è tipografico, il BASIC entra in modalità di edizione presentando la linea che ha provocato l'errore.

L'errore più comune per i "non dattilografi" è Syntax error (il numero 2), ed il BASIC riporta la linea che ha dato problemi se essa è stata incontrata in modo programma. In modo diretto, stabilisce semplicemente che è capitato un errore ed assume che l'ultima linea sia ancora visibile e sia possibile individuare il problema.

Se all'inizio del programma viene incluso un ON ERROR GOTO, è possibile far proseguire l'esecuzione se il computer ha rilevato un errore, ad una data linea. Nell'esempio seguente, il computer, se trova un errore va alla linea 1000

```
10 ON ERROR GOTO 1000
       programma

1000 PRINT CHR$(7):MODE 2:INK 1,0:INK 0,9:CLS:LIST
```

In questo caso il CPC464 emette un bip, cancella lo schermo, cambia la combinazione dei colori ad una utilizzabile in modo ad 80 colonne e lista il programma per permetterne l'esame. Se l'errore è Syntax error, la linea apparirà sotto al listato ad attendere una correzione anche se il messaggio Syntax error viene soppresso.

Si ricordi di terminare con END il programma prima della linea 1000, se si vuole mantenere ciò che è stato visualizzato sullo schermo.

Il BASIC non produce messaggi di errore se le linee sono corrette; perciò, se si riscontra un errore, esso dovrà essere ricercato all'interno delle istruzioni del programma aiutandosi con il messaggio prodotto. Come in molti altri campi è più facile imparare dai propri errori; si facciano esperimenti con il CPC464 che è il più paziente degli insegnanti: ci si stancherà di sbagliare, prima che il CPC464 perda la pazienza!

Tutti gli errori generati dal BASIC sono elencati qui in ordine di numero. I messaggi prodotti dal BASIC sono affiancati da una descrizione delle possibili cause.

1 Unexpected NEXT

E' stato trovato un comando NEXT che non ha un corrispondente FOR, o la variabile di controllo nel comando NEXT non è uguale a quella del FOR.

2 Syntax Error

Il BASIC non comprende la linea in questione che contiene un costrutto non legale.

3 Unexpected RETURN

E' stato trovato un RETURN all'esterno di una sub-routine.

4 DATA exhausted

Un comando READ ha cercato di leggere oltre la fine dell'ultimo DATA

5 Improper argument

E' un errore di carattere generale. Il valore dell'argomento della funzione, o un parametro di un comando non è lecito per qualche motivo.

6 Overflow

Il risultato di una operazione aritmetica ha oltrepassato i limiti numerici della macchina. Può essere un overflow in virgola mobile, nel qual caso l'operazione ha fornito un valore più grande di 1.7 E^38 (circa). Altrimenti può essere il risultato di un tentativo fallito di convertire un numero in virgola mobile in un intero a 16 bit con il segno.

7 Memory full

Lo spazio occupato dal programma attualmente in uso o dalle sue variabili è semplicemente troppo grande oppure una struttura di controllo è innestata troppo profondamente (vale per i GOSUB, i WHILE ed i FOR).

Il comando MEMORY può dare questo errore se si cerca di dare alla memoria del BASIC uno spazio troppo piccolo o troppo grosso. Si noti che ad ogni file aperto corrisponde un buffer che può diminuire lo spazio di memoria utilizzabile dal comando MEMORY.

8 Line does not exist

Non si trova la linea a cui si fa riferimento.

9 Subscript out of range

Uno degli indici di un vettore è troppo grande o troppo piccolo.

10 Array already dimensioned

Uno dei vettori in una istruzione DIM è già stato dichiarato.

11 Division by zero

Può capitare nelle divisioni per numeri reali, interi, moduli interi o nelle esponenziali.

12 Invalid direct command

L'ultimo comando immesso non ha senso in modo diretto.

13 Type mismatch

E' stato presentato un valore numerico dove occorreva una stringa o viceversa, oppure è stato trovato da una istruzione READ o INPUT un numero composto irregolarmente.

14 String space full

Sono state create così tante stringhe che non vi più spazio, anche dopo una pulizia della memoria.

15 String too long

La stringaè più lunga di 255 caratteri. Può essere generato se si cerca di riunire insieme più stringhe.

16 String expression too complex

Le espressioni su stringhe possono generare un gran numero di stringhe intermedie. Quando il numero di tali stringhe eccede un limite ragionevole, viene riportato questo messaggio di errore.

17 Cannot CONTinue

Per qualche motivo il programma non può esser fatto ripartire usando CONT. CONT fa ripartire un programma dopo un comando STOP, dopo una sequenza **[ESC][ESC]** o dopo un errore e che ogni modifica apportata al programma rende il riavvio impossibile.

18 Unknown user function

Non vi un DEF FN per la funzione chiamata.

19 RESUME missing

Si incontrata la fine del programma in modo gestione errori (in una routine ON ERROR GOTO).

20 Unexpected RESUME

L'uso di RESUME è lecito solo in modo gestione errori (in una routine ON ERROR GOTO).

21 Direct command found

Caricando un file è stata trovata una linea senza numero di linea.

22 Operand missing

Il BASIC ha incontrato una espressione incompleta.

23 Line too long

Una linea, convertita nel formato interno del BASIC è diventata troppo lunga.

24 EOF met

E' stato fatto un tentativo di leggere oltre la fine di un file.

25 File type error

Il file che è stato letto non è di un tipo utilizzabile. OPENIN può leggere solo file di testo ASCII. Analogamente LOAD, RUN, ecc. possono utilizzare solo file prodotti da un SAVE.

26 Next missing

Non si trova un NEXT che corrisponda ad un comando FOR.

27 File already open

E' stato eseguito un comando OPENIN o OPENOUT prima di chiudere un file già aperto.

28 Unknown command

Il BASIC non riesce a trovare un comando esterno.

29 WEND missing

Non si trova un WEND dopo un comando WHILE.

30 Unexpected WEND

Si incontrato un WEND all'esterno di un ciclo WHILE o un WEND che non corrisponde ad un comando WHILE.

Parole chiave del BASIC

La seguente è una lista di tutte le parole chiave del BASIC del CPC6128. Come tali esse sono riservate e non possono essere usate per i nomi di variabili.

ABS, AFTER, AND, ASC, ATN, AUTO

BIN$, BORDER

CALL, CAT, CHAIN, CHR$, CINT, CLEAR, CLG, CLOSEIN, CLOSEOUT, CLS, CONT, COS, CREAL

DATA, DEF, DEFINT, DEFREAL, DEFSTR, DEG, DELETE, DI, DIM, DRAW, DRAWR

EDIT, EI, ELSE, END, ENT, ENV, EOF, ERASE, ERL, ERR, ERROR, EVERY, EXP

FIX, FN, FOR, FRE

GOSUB, GOTO

HEX$, HIMEM

IF, INK, INKEY, INKEY$, INP, INPUT, INSTR, INT

JOY

KEY

LEFT$, LEN, LET, LINE, LIST, LOAD, LOCATE, LOG, LOG10, LOWER$

MAX,MEMORY,MERGE,MID$,MIN,MOD,MODE,MOVE,MOVER

NEXT,NEW,NOT

ON, ON BREAK, ON ERROR GOTO, ON SQ, OPENIN, OPENOUT, OR, ORIGIN,OUT

PAPER,PEEK,PEN,PI,PLOT,PLOTR,POKE,POS,PRINT

RAD, RANDOMIZE, READ, RELEASE, REM, REMAIN, RENUM, RESTORE,RESUME,RETURN,RIGHT$,RND,ROUND,RUN

SAVE, SGN, SIN, SOUND, SPACE$, SPC, SPEED, SQ, SQR, STEP,STOP,STR$,STRING$,SWAP,SYMBOL

TAB, TAG, TAGOFF, TAN, TEST, TESTR, THEN, TIME, TO, TROFF,TRON

UNT,UPPER$,USING
VAL,VPOS

WAIT,WEND,WHILE,WIDTH,WINDOW,WRITE

XOR,XPOS

YPOS

ZONE

Indice

APPUNTI

APPUNTI

APPUNTI


CPC464M/1088

AMSTRAD

CPC 464